Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Primož Godec

Barvanje črnobelih slik z globokimi modeli

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: prof. dr. Blaž Zupan

Ljubljana, 2017

Zahvala

Zahvaliti se želim mentorju prof. dr. Blažu Zupanu, za vso pomoč in spodbudo pri raziskovanju in izdelavi magistrske naloge. Za sproščeno delovno okolje in prijetno vzdušje se zahvaljujem članom Laboratorija za bioinformatiko. Zahvala gre tudi moji družini, ki me je podpirala in mi stala ob strani skozi celotno študijsko pot. Prav tako bi se zahvalil prijateljem Manci, Luku, Ožboltu, Roku in Denisu, ki so mi pomagali in poskrbeli, da sem z veseljem sedel v šolskih klopeh in včasih naredili učenje prav zabavno. Zelo velika zahvala gre tudi Ani, ki je poskrbela, da so bila ta študijska leta še lepša, me spodbujala, ko ni šlo vse tako kot mora in mi vedno stala ob strani.

Primož Godec, 2017

Ani.

"I dream of you in colors that don't exist."

— Leah Raeder, Cam Girl

Vsebina

Povzetek

Abstract

1 Uvod **1**

2 Pregled področja **5**

2.1 Predstavitev slikovnih podatkov in barvni prostori 5

2.2 Globoke nevronske mreže 10

2.3 Metode za barvanje črno-belih slik 19

3 Barvanje črno-belih slik z globokimi nevronskimi mrežami **25**

3.1 Arhitekture nevronskih mrež 25

3.2 Pristopi z regresijo 29

3.3 Pristopi s klasifikacijo 32

4 Vrednotenje **35**

4.1 Postopek učenja 35

4.2 Barvanje večjih slik 36

4.3 Podatki 37

4.4 Računanje napake 38

4.5 Primerjava pristopov glede na realističnost barvanja 39

5 Rezultati in diskusija **43**

5.1 Primerjava pristopov na manjši učni množici 43

5.2 Primerjava pristopov na večji učni množici 49
5.3 Primerjava pristopov glede na realističnost barvanja 55
5.4 Barvanje večjih slik 57
5.5 Barvanje starih slik 57
5.6 Konvergenca nevronske mreže 58
5.7 Pomen nivojev mreže 61

6 Zaključek **65**

A Spearmanova korelacija rangov med pristopi **73**

B Implementacije globokih mrež **75**
B.1 Arhitektura S+G 75
B.2 Arhitektura D+G 77
B.3 Arhitektura X-VGG 79

C Primerjava pristopov s spletno anketo **81**

Povzetek

Barvna fotografija je prišla v vsakdanjo uporabo šele v zadnjih 50 letih, zato se v arhivih še vedno nahaja veliko črno-belih fotografij, katere bi njihovi lastniki radi obarvali. V ta namen so bili razviti različni algoritmični pristopi. V disertaciji predstavljamo nekaj novih avtomatskih pristopov za barvanje črno-belih slik in videov, ki so osnovani na strojnem učenju in konvolucijskih nevronskih mrežah. Pristope primerjamo s pristopi iz sorodnih del in jih preizkusimo na starih črno-belih slikah. Iz rezultatov je razvidno, da naši pristopi dosegajo kvaliteto barvanja pristopov iz sorodnih del. Naš nov pristop, ki obarva slike po delih, pa izboljša barvanje slik velikosti, ki so različne od tistih, na katerih je bila mreža naučena. Ta pristop je tudi naučen hitreje kot obstoječi pristopi, ki za barvanje uporabljajo celotne slike.

Ključne besede

umetna inteligenca, strojno učenje, globoke nevronske mreže, barvanje črno-belih slik

Abstract

Title: Deep models for image coloring

Since the color photography came into everyday use in the last fifty years our grandparents are still owning many black and white photographs which we would like to colorize. Researchers are therefore encouraged to develop algorithmic approaches for black and white photographs and video colorization. We have developed a set of automatic approaches based on the machine learning and neural networks, which are using regression and classification. We compared them with approaches from related work. Our approaches reach the quality of colorization comparable to those from related works. Our new approach on image parts improves colorization of images which size is different from those from the training set. This approach is also faster in training than existing approaches that uses full images for learing.

Keywords

artificial inteligence, machine learning, deep neural networks, image colorization

Poglavje 1

Uvod

Čeprav so prvo barvno fotografijo naredili že leta 1886[1], se je ta tehnika v vsakdanji uporabi uveljavila šele mnogo let pozneje. Tako imajo naši stari starši še vedno veliko črno-belih fotografij. Ker te prikazujejo realnost drugače, kot smo navajeni danes, nas zanimajo računski in algoritmični postopki, ki bi znali avtomatsko obarvati črno-bele slike.

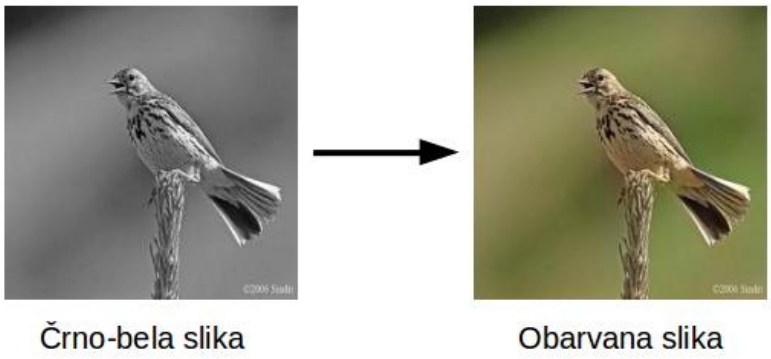
Barvanja črno-belih fotografij so se lotevali že v devetnajstem stoletju, ko so to počeli še ročno. V sedemdesetih letih prejšnjega stoletja so se pojavili prvi pristopi z računalnikom[2], ki so še vedno zahtevali nekaj uporabnikovega sodelovanja. Kasneje so se pristopi izboljševali in postajali vse bolj avtomatski[3] [1, 2, 3, 4, 5].

Algoritmi za barvanje črno-belih slik dobijo kot vhod črno-belo sliko, ki ji dodajo barvo, kot je prikazano na sliki 1.1. Pristopi za barvanje črno-belih slik se uporabljajo na več področjih: barvanje starih slik, barvanje črno-belih filmov in v umetnosti. V preteklosti so bili pristopi za barvanje slik polavtomatski [1, 2, 3, 4, 5], danes pa jih zamenjujejo pristopi, ki obarvajo sliko popolnoma samostojno [6, 7, 8, 9, 10]. Zadnje raziskujemo v tem magistrskem delu.

[1] https://petapixel.com/2015/10/11/a-brief-history-of-color-photography-from-dream-to-reality/

[2] http://articles.chicagotribune.com/1986-08-29/entertainment/8603050091_1_wilson-markle-constance-bennett-laurel-and-hardy-movie

[3] http://www.museum.tv/eotv/colorization.htm

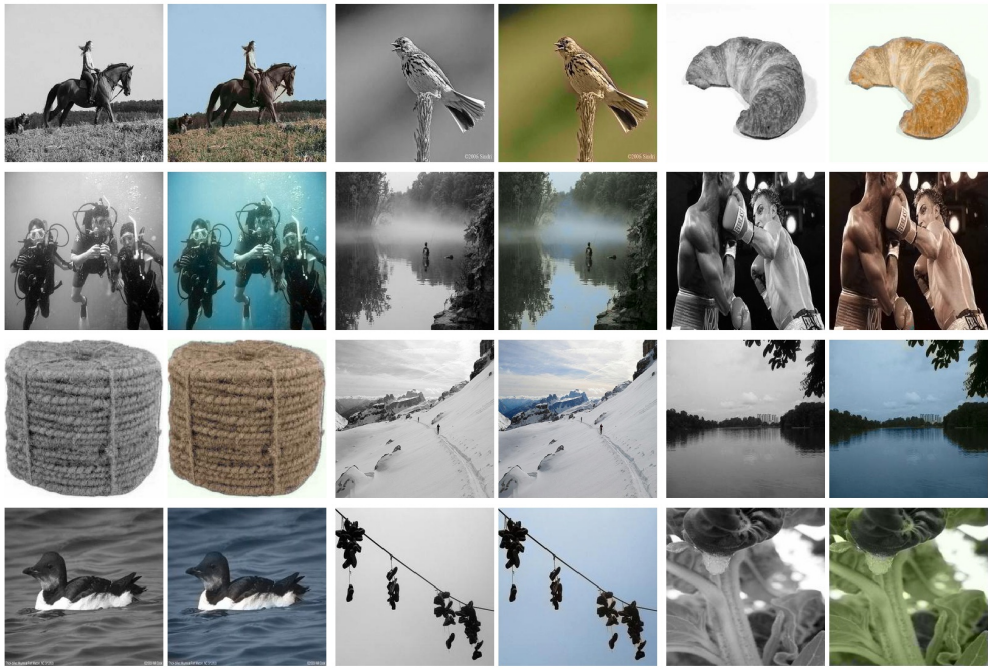


Slika 1.1: Pristop barvanja črno-belih slik za vhod vzame sivinsko sliko (levo) in na izhodu vrne obarvano sliko (desno).

Za človeka je barvanje črno-belih slik, ki so prikazane na sliki 1.2, enostavna naloga. Z vsakdanjim opazovanjem sveta se je človek naučil, da je nebo modro z belimi oblaki, drevesa so zelena in cesta je siva. Za objekte, ki nimajo enolično določene barve, ljudje lahko uganemo, kakšne barve naj bi bili. Pri tem opravilu je potrebno sliko razumeti oziroma prepoznati objekte na sliki, saj iz sivinskih slik ni možno neposredno razbrati barv. Pri nastanku sivinskih slik se dve od treh dimenzij izgubita, saj je barvna slika zapisana z tremi barvnimi kanali v praktično kateremkoli barvnem prostoru, črno-bela pa le z enim barvnim kanalom, ki predstavlja sivino.

Problem barvanja postane bolj kompleksen, ko ga želimo rešiti na avtomatski način z računalnikom. Pri tem so nam v pomoč teksture in njihove lastne barve [8]. Pri objektih, ki nimajo enolično določenih barv — na primer avtomobili, stavbe in knjige — je izziv mnogo težji. Pomaga nam dejstvo, da je naš cilj ohranjanje naravnega izgleda slike in ne reprodukcija originalnih barv. Nihče ne bo vedel, da je bil avtomobil, ki smo ga pobarvali modro, v resnici rdeče barve.

Za barvanje slik smo v sklopu pričujoče naloge implementirali pristope, ki uporabljajo nevronske mreže. Te delujejo podobno kot človeški možgani.



Slika 1.2: Primeri barvanja črno-belih slik. Barvali smo s pristopi predlaganimi v tej nalogi. Za vsako sliko je prikazana črno-bela slika (levo), ki je vhod v algoritem in obarvana slika - izhod algoritma (desno).

Na začetku jih naučimo tako, da jim podamo čim več parov sivinske in ustrezne barvne slike, nevronska mreža pa naučeno znanje oziroma funkcijo, ki sivinsko pretvori v barvno sliko, uporabi za barvanje slik. Za učenje modela potrebujemo veliko črno-belih slik z referenčno barvno sliko. Vzamemo lahko katerokoli barvno sliko in jo pretvorimo v sivinsko.

V nalogi rešujemo problem barvanja črno-belih slik z implementacijo različnih pristopov, ki temeljijo na globokih nevronskih mrežah. Rezultate smo ovrednotili s primerjavo med obarvanimi in originalnimi barvnimi slikami. Rezultate lastnih v nalogi razvitih pristopov smo primerjali s tremi pristopi iz nedavno objavljenih sorodnih del. Ker v naravi obstaja veliko objektov, ki nimajo enolične barve in je naš namen naravnost rezultatov, smo kvaliteto barvanja slik dodatno ocenjevali s spletno anketo.

V nadaljevanju najprej pogledamo sorodna dela, nekaj ozadja o podatkih

in globokih nevronskih mrežah. V tretjem poglavju so podrobno opisane arhitekture nevronskih mrež in predlagani različni pristopi za barvanje. V četrtem poglavju je opisano učenje nevronskih mrež, podatki, na katerih smo te učili in način evalvacije. V petem poglavju smo primerjali naše pristope s tistimi iz sorodnih del in si pogledali, kateri pristopi in slike pri barvanju najbolj izstopajo. Preizkusili smo, kateri pristop deluje najbolje na slikah večjih velikosti in ustreznost barvanja preverili s spletno anketo.

Poglavje 2

Pregled področja

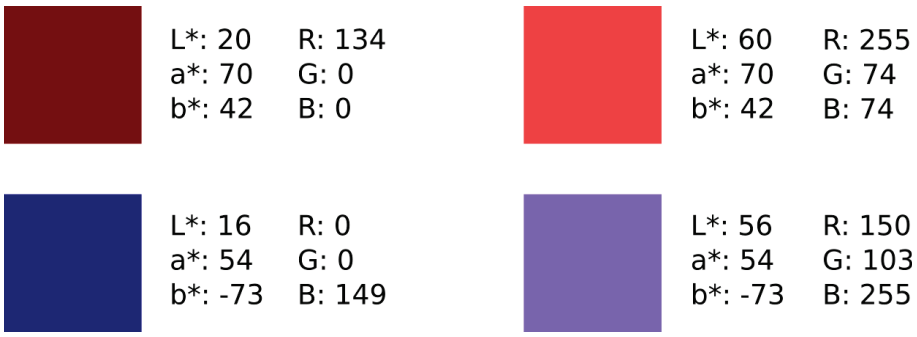
Za barvanje črno-belih slik so bili razviti konceptualno različni pristopi. Nedavno predlagane tehnike uporabljajo globoke nevronske mreže. Pri vseh pristopih barvanja je pomembna tudi predstavitev slikovnih podatkov.

2.1 Predstavitev slikovnih podatkov in barvni prostori

Barvne slike, ki smo jih v disertaciji uporabili pri učenju, so shranjene v barvnem prostoru RGB [11]. Kot je pokazano v [10], se izkaže, da barvanje na osnovi prostora RGB daje slabše rezultate, saj prostor iz dveh razlogov ni primeren za učenje barvanja:

- **Sistem RGB se slabo ujema s človeško percepcijo barv**, saj so številčne razdalje med enako sorodnimi barvami za človeka v prostoru RGB različne [12]. Primer parov dveh barv je predstavljen na sliki 2.1. Različnost med barvami v prvi vrstici je za človekov vizualni sistem enaka različnosti med barvami v drugi vrstici. Razdalja med barvami prvega para v barvnem prostoru RGB je 160, pri drugem paru pa 211.
- **Sistem RGB nima ločenega kanala za svetlost** [11], kar je razvidno iz slike 2.1. Sprememba svetlosti pri barvah v paru povzroči

spremembo vseh komponent v barvnem prostoru RGB. Glede na to, da modeli za barvanje napovedujejo le barvne elemente v sliki, svetlost pa privzamejo iz originalne slike, je najbolj priročno, če uporabljamo barvni prostor, ki ima ločen kanal za svetlost, saj je izhod metode združena komponenta za svetlost z barvnimi komponentami.



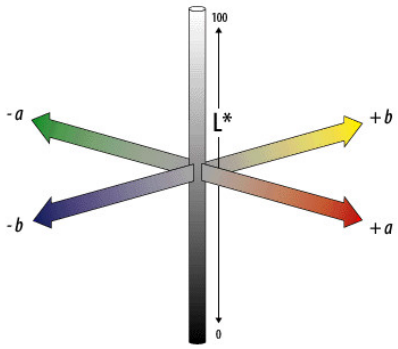
Slika 2.1: Slika prikazuje dva para za ljudi enako oddaljenih barv. Barve so zapisane v barvnih prostorih RGB in CIE L*a*b*. L*a*b* je barvni prostor, kjer enaka številčna razdalja med dvema odtenkoma (na sliki je v obeh primerih razdalja 40) pomeni tudi enako razliko za človekov vizualni sistem [12].

2.1.1 Izbira primernega barvnega prostora

Na podlagi predpostavk o primernosti barvnega prostora je izbira prostorov omejena na Lab [13], YUV [14] in HSL [11]. Vsi našteti prostori vsebujejo ločen kanal za svetlost. Med njimi je Lab edini, kjer bližina barv v prostoru ustreza tudi percepcijski bližini.

Obstaja več implementacij barvnega prostora Lab, ki dobro aproksimirajo človeški zaznavni sistem. Enaka barva je v različnih implementacijah prostora Lab opisana z nekoliko drugačnimi vrednostmi, saj se je prostor razvijal skozi čas in postajal vedno bolj natančen. Trenutno se največ uporablja CIE L*a*b*, ki naj bi bil najboljša aproksimacija človeškega vizualnega sistema [12]. Ta prostor je tudi neodvisen od naprave, ki je sliko zajela.

Prostor CIE L*a*b* predstavi vse barve, ki jih je možno zaznati s tremi barvnimi kanali. Vrednost $L*$ predstavlja svetlost, $a*$ se razširja od zelene proti rdeči in $b*$ od modre proti rumeni barvi (slika 2.2). Zaloga vrednosti $L*$ je od 0, ki predstavlja črno barvo, do 100, ki predstavlja belo barvo [15]. Vrednosti komponent $a*$ in $b*$ načeloma niso omejene, vendar sta v implementacijah omejeni na vrednosti v intervalu $[-128, 127]$, kar je možno predstaviti z osem bitnim celim številom[1]. Ker zaradi pretvorbe iz barvnega prostora RGB redko dosežemo vrednosti komponent $a*$ in $b*$, ki so višje od 100 ali nižje -100, smo opazili, da nekatere implementacije omejijo zalogo vrednosti barvnih komponent na interval $[-100, 100]$. Za pomoč pri implementaciji nevronske mreže smo sami preizkusili, kakšen je dejanski interval barv pretvorjenih iz barvnega prostora RGB (tabela 2.1).



Slika 2.2: Kanali barvnega prostora CIE L*a*b*. $L*$ predstavlja svetlost, $a*$ se razteza od zelene barve v najbolj negativni točki proti rdeči barvi, $b*$ pa se razteza od modre proti rumeni. Nasprotujoče barve na kanalih $a*$ in $b*$ se nikoli ne kombinirajo v odtenek. Vir slike: Adobe, Technical Guid, CIELAB[2].

[1] https://www.freiefarbe.de/en/grenzen-des-cielab-farbraums/
[2] http://dba.med.sc.edu/price/irf/Adobe_tg/models/cielab.html

Tabela 2.1: Največje in najmanjše vrednosti posamezne komponente barvnega prostora CIE L*a*b* pri pretvorbi barv iz barvnega prostora RGB. Pretvorba je bila narejena z uporabo osvetlitve $D65$, ki določa temperaturo bele točke. Izkaže se, da je večina vrednosti komponent $a*b*$ znotraj intervala $[-100, 100]$.

Kanal	Najmanjša vrednost	Največja vrednost
L*	0	100
a*	-86,185	98,254
b*	-107,863	94,482

2.1.2 Pretvarjanje med barvnima prostoroma RGB in CIE L*a*b*

Za pretvorbo med prostoroma ni enostavne enačbe, saj je barvni prostor RGB odvisen od naprave, ki sliko zajame, CIE L*a*b* pa je neodvisen. V našem primeru imamo slike že zapisane v standardnem RGB ali sRGB barvnem prostru, ki je neodvisen od naprave. V ta prostor je bila slika pretvorjena že ob zajemu. Pretvorba v L*a*b* se zato zgodi v dveh korakih [16]:

1. **Pretvorba v barvni prostor CIE 1931** [17] ali drugače imenovan barvni prostor CIE XYZ. Ta pretvorba se izvede s pomočjo linearne pretvorbe - množenje z matriko. Matrika je odvisna od izbire referenčne bele barve [17]. Običajno se izbere referenčno temperaturo bele točke *D65*, ki je tudi standardizirana[3].

2. **Pretvorba iz CIE XYZ v L*a*b*** se izvede z uporabo transformacijskih enačb opisanih v članku XYZ to LAB[4].

Pretvorbo med barvnima prostoroma sRGB in CIE L*a*b* demonstrira sledeča implementacija v Pythonu.

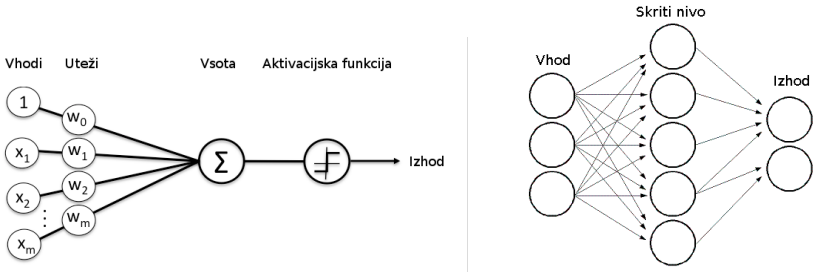
[3]Zapis na uradni strani komisije International Commision on Illumination (krajše CIE), ki je postavila standard, pravi, da se kot standardno uporablja referenčno temperaturo bele točke D65: `http://cie.co.at/index.php?i_ca_id=484`

[4]`http://www.brucelindbloom.com/index.html?Eqn_XYZ_to_Lab.html`

```python
import numpy as np

# matrika za pretvorbo iz sRGB v CIE XYZ
M_srgb_to_xyz = np.array(
    [[0.4124564,  0.3575761,  0.1804375],
     [0.2126729,  0.7151522,  0.0721750],
     [0.0193339,  0.1191920,  0.9503041]])

# referenčna bela barva za osvetlitev D65
xyz_ref_white = np.array([0.95047, 1., 1.08883])

def rgb_to_xyz(rgb):
    _rgb = rgb.copy()
    mask = _rgb > 0.04045
    _rgb[mask] = np.power((_rgb[mask] + 0.055) / 1.055, 2.4)
    _rgb[~mask] /= 12.92
    return _rgb.dot(M_srgb_to_xyz.T)

def xyz_to_lab(rgb):
    _rgb = rgb.copy()
    _rgb /= xyz_ref_white

    mask = _rgb > 0.008856
    _rgb[mask] = np.power(_rgb[mask], 1. / 3.)
    _rgb[~mask] = 7.787 * _rgb[~mask] + 16. / 116.

    L = (116. * _rgb[1]) - 16.
    a = 500.0 * (_rgb[0] - _rgb[1])
    b = 200.0 * (_rgb[1] - _rgb[2])

    return np.array([L, a, b])

def rgb_to_lab(rgb):
    return xyz_to_lab(rgb_to_xyz(rgb))
```

2.2 Globoke nevronske mreže

Nevronska mreža je funkcija $f(\vec{x})$, ki preslika vhod $\vec{x}$ v izhod $\hat{y}$. Med postopkom učenja je ta funkcija optimizirana tako, da najbolje aproksimra dejanske vrednosti $\vec{y}$, ki so za učne podatke znani[5]. Nevronska mreža je sestavljena iz več nivojev. Nivoje si lahko predstavljamo kot vrsto vozlišč, ki se odzovejo v primeru dovolj visokih signalov na vhodu. Primer strukture vozlišča in nivojev je predstavljena na sliki 2.3. Vozlišče pomnoži vhodne vrednosti s trenutnimi vrednostmi uteži, doda še pristranskost (ang. *bias*), vrednosti sešteje in moč aktivacije izračuna s pomočjo tako imenovane aktivacijske funkcije, ki tvori izhod vozlišča. Aktivacijski funkcija vnese nelinearnost, saj poskrbi za to, da nevronska mreža ni le linearna funkcija. Tipični primeri aktivacijskih funkcij so sigmoidna funkcija, tanh, ReLU, leaky ReLU in maxout[6]. Uteži se skozi postopek učenja spreminjajo in s tem določajo aktivacijo vozlišča.



Slika 2.3: Leva slika prikazuje zgradbo enega vozlišča nevronske mreže. Vhod je lahko izhod prejšnjega nivoja ali predstavlja vhodne podatke, ki se potem pomnožijo z utežmi in seštejejo. Desna slika prikazuje zgradbo večnivojske nevronske mreže. V našem primeru ima ta en vhodni nivo, en skriti nivo in izhodni nivo. Vir slike: Introduction to Deep Neural Networks[5] in Neural Networks[7].

Nivojem v nevronskih mrežah, ki se nahajajo med vhodnim in izhodnim

[5] `https://deeplearning4j.org/neuralnet-overview`

[6] A. Karpathy, "Neural Networks Part 1: Setting up the Architecture," Zapiski predavanj, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, 2016.

[7] `http://docs.opencv.org/2.4/modules/ml/doc/neural_networks.html`

nivojem, pravimo skriti nivoji (ang. *hidden layers*) [8]. Tradicionalni algoritmi na področju strojnega učenja so sestavljeni iz vhodnega, izhodnega nivoja in enega skritega nivoja, globoka nevronska mreža (ang. *deep neural network*) pa ima vsaj dva skrita nivoja [18], pri večini praktičnih implementacij pa jih je mnogo več. Vsak nivo globoke nevronske mreže prepozna določene lastnosti vhodnih podatkov. Nivoji, ki se nahajajo globje, lahko prepoznajo bolj kompleksne lastnosti podatkov, saj na vhodu dobijo značilke, ki še dodatno opisujejo vhodne podatke.

Da nevronska mreža daje zadovoljive rezultate, je potrebno utežem določiti ustrezne vrednosti. To naredimo z učenjem. Vsaka nevronska mreža ima cenilno funkcijo (ang. *loss function*), ki pove, kako dobre rezultate daje nevronska mreža na testnih podatkih. V postopku učenja je cilj zmanjšati vrednost cenilne funkcije z enim od algoritmov optimizacije. Tipične cenilne funkcije so cenilna funkcija večrazredne metode podpornih vektorjev (ang. *Multiclass Support Vector Machine Loss*), križna entropija, norma L2 (povprečna kvadratna napaka) in norma L1[9].

2.2.1 Konvolucijske nevronske mreže

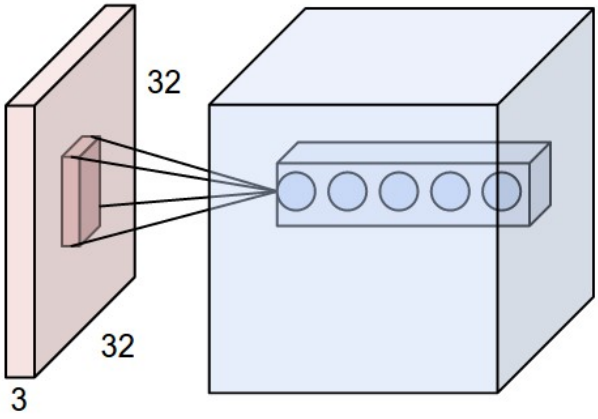
Ker bi bilo na primeru slik pri uporabi klasičnih nevronskih mrež hitro preveč parametrov, kar bi poleg podaljšanja časa učenja povzročilo tudi prekomerno prilagajanje (ang. *overfitting*) in pomanjkanje pomnilnika, uporabljamo konvolucijske nevronske mreže. Te so tako kot običajne nevronske mreže sestavljene iz nevronov, ki imajo svoje uteži in pristranskost. Ti parametri so učljivi. Operacije znotraj nevrona so podobne tistim pri običajnih nevronskih mrežah, le da so prilagojene slikam. Vhod v vsak nivo nevronske mreže je tenzor X z obliko $višina \times širina \times globina$, kjer globina pomeni število

[8] A. Karpathy, "Neural Networks Part 1: Setting up the Architecture," Zapiski predavanj, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, 2016.

[9] A. Karpathy, "Setting up the data and the model," Zapiski predavanj, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, 2016.

barvnih kanalov slike[10]. Konvolucijske nevronske mreže so v osnovi sestavljene iz treh vrst nivojev:

- **Konvolucijski nivo** je glavni gradnik konvolucijske nevronske mreže. Parametri tega nivoja so sestavljeni iz majhnih konvolucijskih jeder, ki pokrivajo majhno polje slike. Več takih jeder pa pokriva celotni nivo v globino, tako si lahko eno jedro predstavljamo kot utež pri skritih nivojih v običajni nevronski mreži, ki zraven izvaja še konvolucijo. Med prehodom po nevronski mreži izvedemo konvolucijo po višini in širini vhodnega tenzorja, po globini pa se izhode teh konvolucij sešteje enako kot pri običajni nevronski mreži. Izhod konvolucije z enim setom jeder je dvodimenzionalna matrika [19]. Na sliki 2.4 je shema, ki prikazuje rezultat konvolucije.



Slika 2.4: Primer, ki prikazuje potek konvolucije. Vhodni tenzor je po celotni višini in širini konvuliran z množico konvolucijskih jeder. Izhod ene operacije je enodimenzionalni tenzor, v tem primeru ima ta pet elementov in vpliva le na del izhodnega tenzorja. Skupek vseh konvolucij tvori celoten izhodni tenzor. Vir slike: Karpathy[10].

- **Nivo združevanja (ang. *pooling layer*)** je namenjen podvzorčenju (ang. *downsampling*) tenzorja. To izvede tako, da združi več izhodov

[10] A. Karpathy, "Understanding and Visualizing Convolutional Neural Networks," Zapiski predavanj, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, 2016.

iz nevronov nekega nivoja v posamezen nevron v naslednjem nivoju. Za združevanje obstaja več strategij. Najbolj pogosto je maksimalno združevanje (ang. *max pooling*), poznamo pa še povprečno združevanje (ang. *average pooling*). Z združevanjem zmanjšamo prostorsko dimenzijo tenzorja in s tem število parametrov, kar vpliva na zmanjšanje računske zahtevnosti in prekomernega prilagajanja. Deluje na principu, da je točna lokacija značilke manj pomembna kot približna lokacija glede na ostale značilke. [20].

- **Polnopovezni nivo** je nivo enak skritim nivojem pri klasični nevronski mreži. Večinoma se uporabi za zadnjih nekaj nivojev pri konvolucijski nevronski mreži.

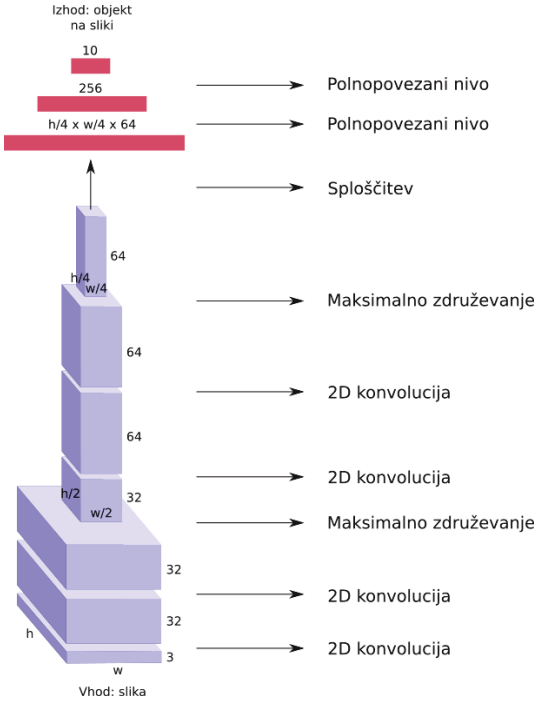
2.2.2 Primer konvolucijske nevronske mreže

Za primer konvolucijske nevronske mreže bomo predstavili poenostavljeno mrežo za prepoznavanje objektov v sliki. Ta naloga je namreč postala šolski primer uporabe konvolucijskih nevronskih mrež [21].

Odločili smo se implementirati mrežo, ki na vhodu prejme sliko velikosti $h \times w$ slikovnih točk in ima tri barvne kanale prostora RGB. Mreža na izhodu vsako sliko klasificira v enega od razredov. Za demonstracijo smo se odločili, da bomo slike klasificirali v enega od desetih razredov, čeprav večina ostalih implementacij slike razvršča v mnogo več razredov. Primeri razredov so na primer živali, drevesa, zgradbe, morje in ljudje in so odvisni od podatkovne zbirke, s katero učimo mrežo.

Mreža, ki jo uporabljamo za to nalogo, je shematsko prikazana na sliki 2.5. Na začetku ima dva konvolucijska nivoja, nato sledi nivo maksimalnega združevanja, ki zmanjša prostorsko dimenzijo tenzorja. Sledita še dve konvoluciji in še eno maksimalno združevanje. Takoj za tem se tenzor splošči v enodimenzionalni tenzor in gre preko dveh polnopovezanih nivojev. Prvi velikost enodimenzionalnega tenzorja zmanjša na velikost 256, drugi pa na velikost 10. Vsaka vrednost v zadnjem tenzorju predstavlja verjetnost za pri-

padnost objekta na sliki določenemu razredu. Razred z največjo verjetnostjo predstavlja objekt na sliki.



Slika 2.5: Shematski prikaz poenostavljene nevronske mreže za prepoznavanje objektov na sliki. Na levi strani je prikazano spreminjanje velikosti tenzorjev ob prehodu skozi nevronsko mrežo, na desni pa so označeni nivoji mreže.

Vsak nivo ima tudi svojo aktivacijsko funkcijo. Pri vseh nivojih uporabljamo funkcijo ReLU, razen pri zadnjem, polnopovezanem, kjer je aktivacijska funkcija softmax. Glede na to, da je problem klasifikacijski, za cenilno funkcijo uporabljamo križno entropijo. Primer implementacije te nevronske mreže v vmesniku Keras je predstavljen v nadaljevanju.

2.2.3 Konvolucijske nevronske mreže v Kerasu

Keras[11] je visokonivojski programski vmesnik za nevronske mreže. Vmesnik lahko teče na zaledju Tensorflow [22], Theano [23] ali CNTK [24]. V našem

[11] `https://github.com/fchollet/keras`

primeru smo izbrali Tensorflow. Razlog za izbiro vmesnika Keras je, da je visokonivojski in zaradi tega uporabniku prijazen, narejen je v programskem jeziku Python[12], ki nam je najbolje poznan in je zelo modularen, nevronska mreža je predstavljena kot sekvenca ali graf, ki jo je enostavno dopolniti ali spremeniti.

V spodnji kodi smo implementirali konvolucijsko nevronsko mrežo iz poglavja 2.2.2.

```
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.optimizers import Adam

""" Priprava naključnih podatkov """
x_train = np.random.random((100, 100, 100, 3))
y_train = keras.utils.to_categorical(
    np.random.randint(10, size=(100, 1)), num_classes=10)
x_validation = np.random.random((20, 100, 100, 3))
y_validation = keras.utils.to_categorical(
    np.random.randint(10, size=(20, 1)), num_classes=10)
x_test = np.random.random((20, 100, 100, 3))

""" Arhitektura modela """
model = Sequential()

model.add(
    Conv2D(32, (3, 3), activation='relu', input_shape=(100, 100, 3)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

[12] https://www.python.org/

```
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))

""" Učenje modela """
adam = Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-08)
model.compile(loss='categorical_crossentropy', optimizer=adam)

model.fit(x_train, y_train, batch_size=32, epochs=10)

""" Validacija modela """
score = model.evaluate(x_test, y_test, batch_size=32)
```

Kot je razvidno iz prvega dela implementacije, so podatki, ki smo jih pripravili, generirani naključno. Ti podatki nam sicer ne zagotavljajo smiselnega učenja, iz njih pa lepo vidimo strukturo podatkov. Pri podatkih predpona `x` označuje, da gre za vhodne podatke v mrežo, `y` označuje prave vrednosti oziroma pričakovane izhode iz mreže. Običajno razlikujemo med tremi vrstami podatkov. Učni podatki (`x_training`, `y_training`) so tisti, ki jih uporabimo zgolj za učenje mreže, validacijski podatki (`x_validation`, `y_validation`) so tisti, ki imajo tako kot učni podatki zraven prisotne prave vrednosti in služijo preverjanju natančnosti mreže. Testni podatki (`x_test`) so tisti, ki jih na koncu uporabimo za prikaz rezultatov mreže. Velikokrat testni podatki nimajo prisotnih pravih vrednosti. Na primer pri raznih tekmovanjih na področju strojnega učenja tekmovalec dobi le vhodne podatke, napovedi pa odda za kasnejše vrednotenje, pri čemer ne vidi pravilnih napovedi.

Mreža ima sekvenčno zgradbo, konvolucijski nivoji uporabljajo jedro velikosti 3×3, pri maksimalnem združevanju (`MaxPooling`) uporabljamo velikost združevanja 2×2, kar pomeni, da se velikost tenzorja po višini in širini zmanjša za faktor polovico. V implementaciji se uporabljajo tudi nivoji za

opustitev nevronov (`Dropout`), ki v vsakem koraku učenja izločijo določen delež vozlišč. Ta vozlišča s tem ne vplivajo na rezultat napovedi in se v tistem koraku tudi ne učijo. S tem zmanjšajo prekomerno prilagajanje (ang. *overfitting*).

Pri učenju smo uporabljali optimizator `Adam` [25]. Optimizator je algoritem, ki določa strategijo spreminjanja uteži in s tem zmanjšuje vrednost cenilne funkcije. Najbolj enostavni optimizator je gradientni sestop, ki spreminja uteži v smeri najhitrejšega zmanjševanja napake, vendar to ni vedno dovolj učinkovito, zato so bili razviti novi algoritmi. V praksi se sicer največ uporabljajo optimizacijski pristopi: SGD, Adagrad, RMSprop in Adam[13].

Kot lahko opazimo ob pogledu na funkcijo `fit`, je parameter `epochs` nastavljen na fiksno število korakov. V tem primeru mrežo učimo z desetimi prehodi skozi vse podatke. Običajno se trajanje učenja določi s spremljanjem obnašanja validacijske napake. Z učenjem končamo, ko se validacijska napaka neha zmanjševati. Takrat pravimo, da je algoritem skonvergiral k najmanjši napaki. Če bi z učenjem nadaljevali, se začne mreža prekomerno prilagajati učnim podatkom, kar v večini primerov pomeni, da se slabše odreže na validacijskih in kasneje testnih podatkih.

2.2.4 Cenilne funkcije

Pri pristopih, ki smo jih opisali v nalogi, uporabljamo tri cenilne funkcije. Pri regresijskih pristopih se je za dobro izkazala povprečna kvadratna napaka (ang. *mean squared error*), ki jo označimo z MSE:

$$\mathrm{MSE}(\hat{Y}, Y) = \frac{1}{hwc} \sum_{i=0}^{h} \sum_{j=0}^{w} \sum_{k=0}^{c} (Y_{i,j,k} - \hat{Y}_{i,j,k})^2. \tag{2.1}$$

Spremenljivka Y predstavlja originalno sliko, $\hat{Y}$ pa obarvano sliko s strani mreže. Spremenljivka h je višina slike, w je širina slike in c je število kanalov.

[13] A. Karpathy, "Learning," Zapiski predavanj, CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, 2016.

MSE smo računali le na komponentah $a*$ in $b*$, tako da je $c = 2$.

Pri enem od klasifikacijskih pristopov smo za cenilno funkcijo uporabili divergenco Kullback-Leibler [26], ki jo označujemo z $\mathrm{D_{KL}}$. V našem primeru smo izvedli divergenco Kullback-Leibler na posamezni slikovni točki in jo povprečili preko celotne slike:

$$\mathrm{D_{KL}}(Z||\hat{Z}) = \frac{1}{hw}\sum_{i=0}^{h}\sum_{j=0}^{w}\sum_{k=0}^{c} Z_{i,j,k}\log\frac{Z_{i,j,k}}{\hat{Z}_{i,j,k}}. \tag{2.2}$$

V enačbi Z predstavlja originalno sliko, $\hat{Z}$ pa obarvano sliko s strani mreže, obe sta pretvorjeni v vektorje s 400 razredi, ki predstavljajo barve. Spremenljivka h je višina slike, w je širina slike, oznaka c v tem primeru predstavlja število razredov, v katere mreža klasificira barve, zato je $c = 400$.

Kot zadnjo cenilno funkcijo smo uporabili tudi križno entropijo (ang. *cross entropy*) [27]. Križna entropija se računa na razredih napovedanih s strani mreže:

$$\mathrm{H}(\hat{Z}, Z) = \sum_{i=0}^{h}\sum_{j=0}^{w}\sum_{k=0}^{c} Z_{i,j,k}\log\hat{Z}_{i,j,k}. \tag{2.3}$$

Druga različica križne entropije uporablja uteži, ki predstavljajo povprečno pogostost pojavitve posamezne barve v slikah. Pogostost je bila izračunana na množici 100.000 naključnih slik izbranih iz zbirke ImageNet. Uteži v cenilni funkciji dajejo večji pomen tistim točkam, kjer se v originalni sliki pojavijo močnejše barve. S tem želimo vzpodbuditi mrežo, da bo večkrat obarvala z močnejšimi barvami. Pri ostalih pristopih je namreč problem, da barve velikokrat niso tako kontrastne, kot bi lahko bile. Križno entropijo z utežmi označimo s Hw, kjer $v(\cdot)$ predstavlja utež za barvo v slikovni točki na koordinatah (h, w):

$$\mathrm{Hw}(\hat{Z}, Z) = \sum_{i=0}^{h}\sum_{j=0}^{w} v(Z_{i,j})\sum_{k=0}^{c} Z_{i,j,k}\log\hat{Z}_{i,j,k}. \tag{2.4}$$

2.3 Metode za barvanje črno-belih slik

Pristope za barvanje črno-belih slik lahko delimo v dve večji skupini. Prva zahteva interakcijo uporabnika med postopkom barvanja, ki se je več uporabljala v preteklosti, pri drugi pa barvanje poteka popolnoma avtomatsko.

2.3.1 Pristopi z interakcijo uporabnika

To skupino pristopov delimo na tehnike, ki temeljijo na uporabnikovem barvanju manjših delov slik (ang. *scribble based*) [1, 2] in tiste, ki temeljijo na primerih podobno obarvanih slik (ang. *example based*) [3, 4, 5]. Pri prvih uporabnik določi barvo nekaj točkam na sliki, te pa algoritem avtomatsko razširi preko cele slike. Kvaliteta barvanja je odvisna od zahtevnosti slike in števila točk, ki jih je uporabnik označil. Pri barvanju na primerih uporabnik izbere referenčno sliko podobno tisti, ki jo želimo obarvati, algoritem pa lastnosti izbrane slike razširi na drugo sliko ali množico slik. Kvaliteta barvanja je odvisna od podobnosti referenčne slike s sliko, ki jo barvamo.

Prednost tehnike barvanja, ki temelji na barvanju manjših delov je, da je barvanje lahko zelo natančno in naravno, vendar mora uporabnik za dovolj veliko natančnost označiti veliko točk na sliki. V nasprotnem primeru je lahko barvanje tudi zelo nenatančno. Prednost tehnike barvanja s primeri pa je njena hitrost in avtomatičnost po tem, ko imamo izbrano referenčno sliko, vendar je velikokrat problem najti dobro referenčno sliko. Zaradi teh lastnosti se tehnika barvanja s primeri uporablja za barvanje videa, saj je v tem primeru potrebno ročno pobarvati na primer vsako stoto sliko, na ostale pa algoritem sam razširi lastnosti ročno barvane slike. Ta tehnika je večinoma zelo natančna, saj se slike, ki so v videu blizu, običajno zelo malo razlikujejo.

2.3.2 Popolnoma avtomatski pristopi

V magistrskem delu se osredotočamo na avtomatske pristope barvanja. Ti pristopi samostojno brez uporabnikovega posredovanja obarvajo celotno sliko.

Prva dva pristopa, ki sta bila predlagana na tem področju, temeljita na značilkah slik, ki opisujejo intenziteto posamezne barve in robove. Prvi pristop uporablja za barvanje nevronsko mrežo [6], ki vsebuje zgolj polnopovezane nivoje, druga pa za barvanje uporabi metodo naključnih gozdov [7].

Novejši pristopi barvanja uporabljajo konvolucijske nevronske mreže, ki v vsakem nivoju same odkrijejo značilke, ki so pomembne za čimbolj kvalitetno barvanje. Prva tovrstna rešitev[14] gradi mrežo na podlagi značilk že zgrajene šestnajst-nivojske mreže VGG-16, ki so jo razvili na univerzi v Oxfordu [28] in je namenjena prepoznavanju objektov na sliki. Rešitev uporablja evklidsko cenilno funkcijo in barvni prostor YUV [14]. Njena slabost je slabša barvna nasičenost in poudarjenost rjavih odtenkov.

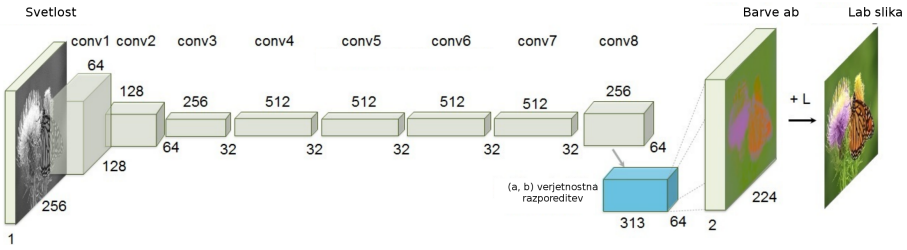
Problem nenasičenosti je moč odpraviti z uporabo softmax funkcije v zadnjem nivoju nevronske mreže. Problem barvanja na ta način spremeni v klasifikacijskega. Zhang in sod. [8] uporabijo konvolucijsko nevronsko mrežo z več nivoji in aktivacijskimi funkcijami ReLU. Uporabljajo barvni prostor L*a*b*. Arhitektura je predstavljena na sliki 2.6. Posebnost te mreže je cenilna funkcija. Uporablja križno entropijo (enačba 2.5), ki je v tem primeru izvedena na primerjavi barv vsake slikovne točke glede na barvni prostor, ki je kvantiziran:

$$H(\hat{Z}, Z) = -\sum_{i=0}^{h}\sum_{j=0}^{w} v(Z_{i,j}) \sum_{k=0}^{c} Z_{i,j,k} \log(\hat{Z}_{i,j,k}). \tag{2.5}$$

Spremenljivka Z predstavlja barve originalne slike, ki so zapisane kot c razsežni vektorji, kjer c predstavlja število barv. $\hat{Z}$ je ocenjena barva. Napake so pomnožene z utežmi, ki odražajo pogostost barv. Redkeje uporabljene barve so utežene tako, da prispevajo večji delež k cenilni funkciji. S tem so avtorji izboljšali rezultate tako, da se bolj pogosto pojavljajo tudi močnejši odtenki, torej tisti z višjimi vrednostmi v barvnem prostoru $a*b*$, ki so bili prej redkeje zastopani zaradi bolj pogostega pojavljanja nežnejših barv v slikah. To so barve bližje vrednostim $(0, 0)$ v prostoru $a*b*$. V enačbi 2.5 za

[14] `http://tinyclouds.org/colorize/`

obteževanje vrednosti poskrbi funkcija $v(\cdot)$. Pogostost je bila izračunana z analizo vseh slik v podatkovni zbirki Imagenet [21].



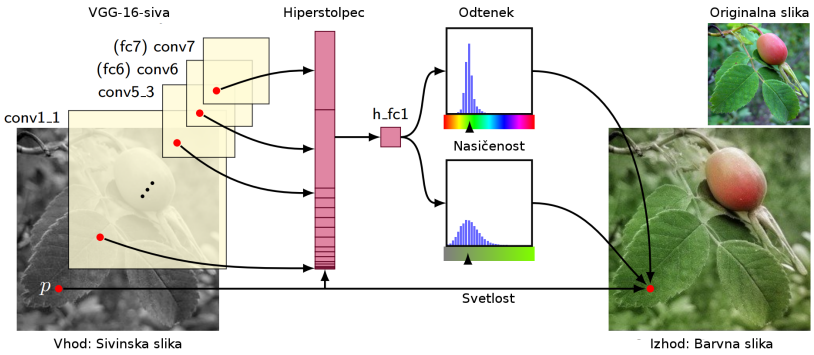
Slika 2.6: Arhitektura mreže Zhang in sod. [8]. Na vhodu je črno-belo slika, nato sledi več konvolucijskih blokov. Vsak od blokov vsebuje dve ali tri konvolucije. Vsaki konvoluciji sledi normalizacija serije in aktivacijska funkcija ReLU. Modri blok predstavlja napovedi barve v obliki verjetnosti za vsak razred, ki predstavlja množico odtenkov. Te vrednosti se na koncu pretvorijo v barvne vrednosti $a * b*$ in združijo s črno-belo sliko, da dobimo obarvano sliko. Vir slike: Zhang in sod. [8].

Larsson in sod. [9] za osnovo uporabijo mrežo VGG-16, iz katere vzamejo tenzorje vsakega nivoja, ki jim povečajo prostorsko dimenzijo tako, da se ujemajo in združijo v enotno matriko. Sledi še en polno-povezan nivo. Rezultat klasifikacije je histogram z verjetnostmi posameznega odtenka za vsako točko na sliki. Arhitektura mreže je predstavljena na sliki 2.7. Uporabljajo barvni prostor HSV, ki ga prilagodijo zaradi nestabilnosti v delu prostora v komponente: odtenek označen z H, barvna nasičenost označena z C in svetlost L. Cenilna funkcija, ki jo uporabljajo, je divergenca Kullback-Leibler, ki primerja izhodni histogram z originalno sliko pretvorjeno v histogram po zgoraj opisanih komponentah:

$$\begin{gathered} L(\hat{Z}, Z) = \sum_{i=0}^{h} \sum_{j=0}^{w} \left(\mathrm{D_{KL}}(Z_{i,j,C} || \hat{Z}_{i,j,C}) + \lambda \mathrm{D_{KL}}(Z_{i,j,H} || \hat{Z}_{i,j,H}) \right) \\ \mathrm{D_{KL}}(z||\hat{z}) = \sum_{i=0}^{c} z_i \log \frac{z_i}{\hat{z}_i}. \end{gathered} \tag{2.6}$$

V enačbi 2.6 Z predstavlja originalno sliko spremenjeno v histogram vredno-

sti za vsako od komponent H in C, $\hat{Z}$ predstavlja ocenjeno barvo. Utež λ določa, koliko k napaki prispeva vsaka od komponent C in H in so jo avtorji nastavili na vrednost $\lambda = 5$.

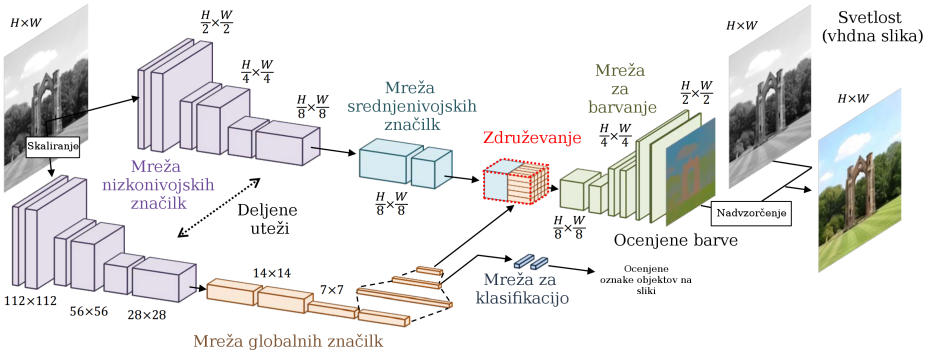


Slika 2.7: Slika prikazuje arhitekturo mreže Larsson in sod. [9]. Mreži VGG-16 je podana črno-belo slika, nato se iz omenjene mreže vzame izhode vseh konvolucij in polnopovezanih nivojev, jih nadvzorči na velikost osnovne slike ter združi v skupni tenzor, kot je prikazano na sliki. Sledi še en polnopovezani nivo in dobimo ocene v obliki histogramov verjetnosti, ki se jih potem pretvori v barvni prostor HSV in združi v sliko skupaj z originalno sliko. Vir slike: Larsson in sod. [9].

Iizuka in sod. [10] uporabijo nevronsko mrežo sestavljeno iz dveh delov (slika 2.8). Del za klasifikacijo poskrbi za napovedovanje vsebine slike, ki se potem združi z glavnim delom in izboljša natančnost barvanja. Uporabljajo barvni prostor L*a*b. Za cenilno funkcijo so uporabili povprečno kvadratno napako v kombinaciji s križno entropijo:

$$L_(\hat{Y}^{color}, Y^{color}, \hat{Y}^{class}, l^{class}) = \sum_{i=0}^{h}\sum_{j=0}^{w}\left(\left\|Y_{i,j}^{color} - \hat{Y}_{i,j}^{color}\right\|_{FRO}^{2} - \lambda\left(\hat{Y}_{l^{class}}^{class} - \log\left(\sum_{i=0}^{N}\exp(\hat{Y}_{i}^{class}\right)\right)\right). \tag{2.7}$$

Prva je namenjena ocenjevanju kvalitete barvanja, druga pa ocenjuje natančnost klasifikacije objektov na sliki. S tem je poskrbljeno, da se skupaj z učenjem glavne mreže uči še mreža za klasifikacijo. Oznaka y^{color} predstavlja barvo slikovne točke na originalni sliki, $\hat{y}^{color}$ ocenjeno barvo, $\hat{y}^{class}$ ocenjen razred, ki določa vsebino slike in l^{class} indeks objekta, ki se v resnici nahaja na sliki. Z $\|\cdot\|_{FRO}$ označimo Frobeniusovo normo in z λ utež, ki določa razmerje med vplivom mreže za klasifikacijo in glavne mreže za učenje. Za razliko od prejšnjih dveh pristopov pristop Iizuka in sod. napoveduje direktno $a*$ in $b*$ vrednosti ter za to uporabi regresijo.



Slika 2.8: Arhitektura mreže Iizuka in sod. [10]. Mreža dobi na vhodu črno belo sliko. Ta se poda tako mreži za klasifikacijo (spodaj), kot mreži za barvanje (zgoraj). Izhod mreže je slika z barvnimi komponentami $a * b*$, ki se nato združi s črno-belo sliko. Vir slike: Iizuka in sod. [10].

Poglavje 3

Barvanje črno-belih slik z globokimi nevronskimi mrežami

V nalogi smo razvili nekaj različnih arhitektur globokih nevronskih mrež. Razvili smo tako pristope z regresijo kot klasifikacijo. Razvili smo tudi pristop, ki zna učinkovito barvati slike večje od teh iz učne množice. Pri arhitekturah mrež smo se zgledovali po sorodnih, že obstoječih pristopih, a v eksperimentih skušali razumeti, kateri elementi teh pristopov najbolj vplivajo na kvaliteto barvanja.

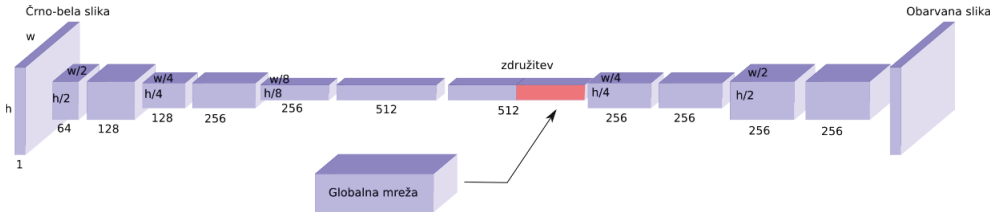
3.1 Arhitekture nevronskih mrež

Implementirali smo štiri arhitekture nevronskih mrež in jih kombinirali z različnimi cenilnimi funkcijami, pristopi (regresijski ali klasifikacijski) in načini napovedovanja (lokalni in globalni).

3.1.1 Arhitektura S+G

Arhitektura S+G (ang. *Shallow with Global*) je sestavljena iz dveh delov, ki se kasneje združita v enotno mrežo. Glavni del predstavlja zaporedje konvolucijskih nivojev, ki na vhodu sprejmejo črno-belo sliko z enim kanalom, izhod pa je obarvana slika z dvema kanaloma za barvni komponenti $a*$ in

$b*$. Po prvih osmih kovolucijskih nivojih se mreža združi s tako imenovano globalno mrežo, ki prepoznava objekte na sliki. Za globalno mrežo smo po vzoru [10] vzeli že naučeno šestnajstnivojsko mrežo VGG-16 [28], ki smo ji odvzeli zadnji polnopovezani nivo in ji dodali nov polnopovezani nivo z izhodnim tenzorjem dolžine 256. Ker je arhitektura VGG-16 namenjena sprejemu barvnih slik s tremi vhodnimi kanali, smo vhod prilagodili tako, da sprejme sivinsko sliko na vsakem od vhodnih kanalov. Arhitektura nevronske mreže je predstavljena na sliki 3.1 in z implementacijo v prilogi B.1.

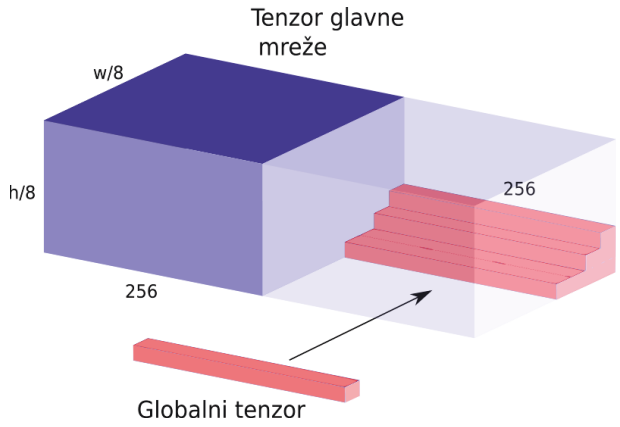


Slika 3.1: Velikosti tenzorjev skozi arhitekturo S+G. Oznaki h in w predstavljata višino in širino vhodne slike. Podrobnosti nivoja združitev so predstavljene na sliki 3.2, nivo izhodne slike ni natančneje označen, saj se razlikuje v različnih implementacijah, ki so podrobneje opisane v poglavjih 3.2 in 3.3.

Vhod v element za združevanje glavne in globalne mreže sta tenzorja velikosti $\frac{h}{8} \times \frac{w}{8} \times 256$ iz glavne mreže in enodimenzionalni tenzor velikosti 256 iz globalne mreže. Pri tem h in w predstavljata višino in širino vhodne slike v mrežo. Pri združevanju vsakemu elementu širine in višine prvega tenzorja pridružimo tenzor globalne mreže (slika 3.2). Tako na izhodu dobimo tenzor velikosti $\frac{h}{8} \times \frac{w}{8} \times 512$.

3.1.2 Arhitektura D+G

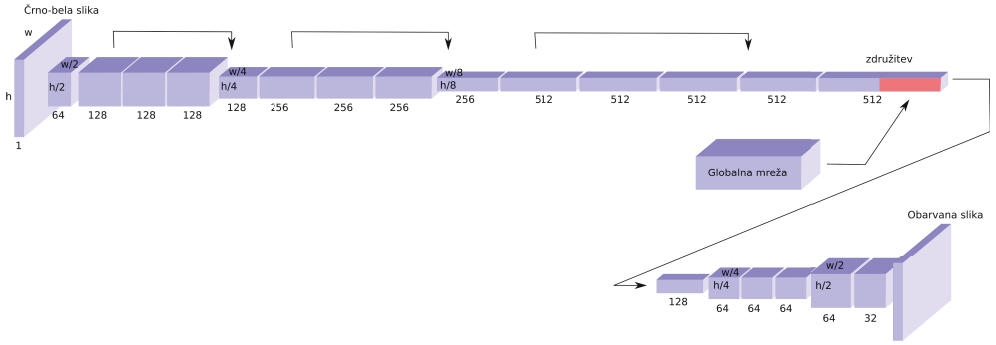
Arhitektura D+G (ang. *Deep with Global*) ima v osnovi enako zasnovo kot S+G, a uporabi 14 konvolucijskih nivojev pred združitvijo in 7 po združitvi (slika 3.3 in priloga B.2). Za zmanjševanje prostorskih dimenzij uporablja maksimalno združevanje [20] in za povečevanje le-teh v zadnjih nivojih transponirano konvolucijo (ang. *transpose convolution*) [29], imenovano tudi



Slika 3.2: Delovanje nivoja združevanja glavne nevronske mreže z globalno nevronsko mrežo. K izhodu glavne nevronske mreže (modri tenzor) dodamo tenzor iz globalne mreže (rdeči tenzor), tako da ga priključimo k vsem prostorskim lokacijam glavnega tenzorja kot dodatnih 256 kanalov. Izhodni tenzor ima tako velikost $\frac{h}{8} \times \frac{w}{8} \times 512$.

dekonvolucija. Transponirana konvolucija je transformacija, ki deluje v nasprotni smeri kot običajna konvolucija. Na vhodu vzame tenzor dimenzije, ki je izhod konvolucije z enakimi parametri, izhod transponirane konvolucije pa je tenzor dimenzije enake tistemu na vhodu običajne konvolucije. Transponirana konvolucija za svoje delovanje še vedno uporablja princip konvolucije. Združevanja glavne in globalne mreže se izvede, kot je opisano v poglavju 3.1.1.

Arhitektura D+G prinaša še eno spremembo. To so tako imenovane rezidualne povezave, ki so bile prvič uporabljene v nevronski mreži ResNet, zasnovani s strani Microsoft Research Asia [30], ki je leta 2015 zmagala na tekmovanju ImageNet [21]. Te povezave so na sliki 3.3 označene s puščicami nad nevronsko mrežo in predstavljajo povezavo, ki na mestu, kamor kaže puščica, združi trenutni tenzor s tenzorjem izračunanim pred dvema nivojema. Operacija združevanja pomeni seštevanje isto ležečih elementov v tenzorju. Za rezidualne povezave smo se odločili, saj naj bi te prisilile nivoje, da se naučijo nekaj novega in ne le nadgradijo izhod prejšnjega nivoja. S tem smo računali na izboljšavo v natančnosti barvanja.



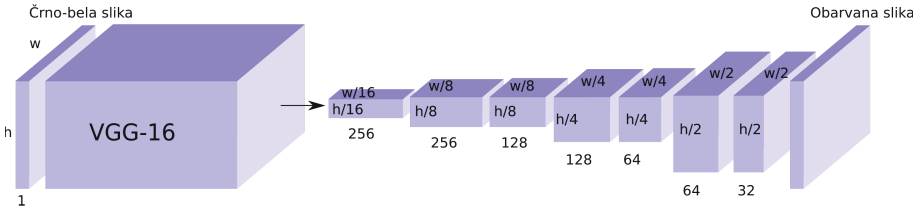
Slika 3.3: Tenzorji v arhitekturi D+G. Oznaki h in w predstavljata višino in širino vhodne slike. Podrobnosti nivoja združitev so predstavljene na sliki 3.2, nivo izhodne slike pa ni natančneje označen, saj je odvisen od implementacije, ki so podrobneje opisane v poglavjih 3.2 in 3.3.

3.1.3 Arhitektura D

Ta arhitektura je v glavnem delu enaka arhitekturi D+G, a ne uporablja globalne mreže. Mreža za vhod vzame črno-belo sliko z enim kanalom in izračuna barvno sliko na izhodu samo preko konvolucijskih nivojev. Arhitekturo D smo uporabili z namenom preverjanja vpliva globalne mreže na rezultate barvanja.

3.1.4 Arhitektura X-VGG

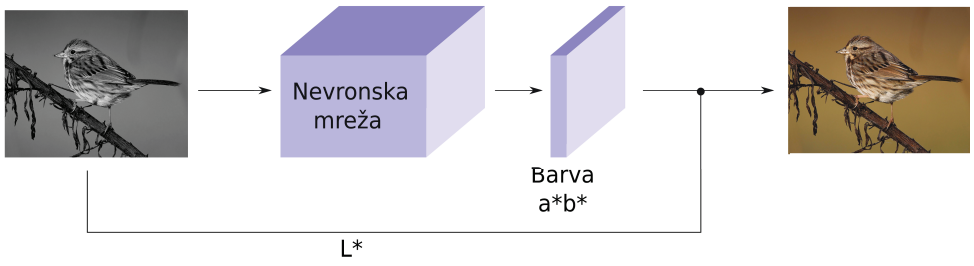
Arhitektura X-VGG je zgrajena tako, da za nižje nivoje uporabimo mrežo VGG-16 [28], kateri smo odstranili vse zadnje polnopovezane nivoje. Na vhodu sprejme črno-belo sliko, ki se nato prilagodi za vhod nevronske mreže VGG-16, tako da ima tri vhodne kanale. Te pridobimo tako, da vzamemo sivinsko sliko za vsak vhodni kanal. Tenzor, ki ga vrne zadnji konvolucijski nivo mreže VGG-16, podamo na vhod prvega nivoja nove mreže (slika 3.4 in priloga B.3). Nova mreža ima dodatnih osem konvolucijskih nivojev in štiri nivoje nadvzorčenja, ki poskrbijo za povečanje dimenzije prostorskih komponent. Po zadnjem nivoju izvedemo še povečanje slike za faktor dve, saj se nadvzorčenje izkaže kot slabše.



Slika 3.4: Velikosti tenzorjev skozi arhitekturo X-VGG. Oznaki h in w predstavljata višino in širino vhodne slike. Nivo izhodne slike ni natančneje označen, saj je odvisen od implementacije (poglavji 3.2 in 3.3). Prvi večji blok predstavlja mrežo VGG-16 [28]

3.2 Pristopi z regresijo

Regresijski pristopi direktno napovejo vrednosti barvnih komponent $a*$ in $b*$ v prostoru CIE L*a*b*. Tu mreža predstavlja regresijsko funkcijo $\vec{y} = f(x)$ za vsako točko slike, ki na vhod dobi točko sivinske slike x, izhod pa sta zvezni vrednosti $a*$ in $b*$ (slika 3.5). Mreži podamo sivinsko sliko, ta oceni barvni komponenti $a*$ in $b*$, nato izhod mreže združimo s sivinsko sliko, ki je obenem komponenta $L*$.



Slika 3.5: Regresijski pristopi na vhod prejmejo črno-belo sliko in s pomočjo nevronske mreže izračunajo barvni komponenti $a*$ in $b*$ barvnega prostora *CIE L*a*b**.

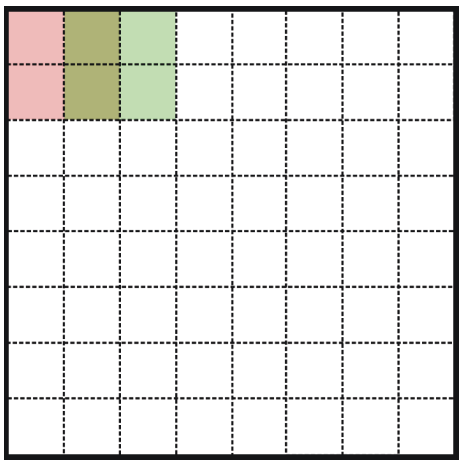
3.2.1 Lokalni pristop

Barvanje pri tem pristopu izvedemo ločeno na majhnih delih slik, kot človek, ki bi na primer ločeno obarval vodo, nato gozd in kasneje še nebo.

Sliko razdelilmo na koščke velikosti 32×32 slikovnih točk. Pri tem se sosednja koščka prekrivata za 16 slikovnih točk, kot je prikazano na sliki 3.6. V arhitekturah, kjer je prisotna ločena globalna mreža, ta še vedno na vhodu prejme celotno sliko. Dele slik po barvanju sestavimo z metodo prekrivanja, tako da imajo vrednosti točk pri robu manjši vpliv kot tiste pri sredini. Vpliv barvne točke se izračuna po enačbi

$$Y'_{h,w} = \frac{d}{16} Y_{h,w}, \tag{3.1}$$

kjer $Y_{h,w}$ predstavlja vrednost slikovne točke, d pa predstavlja oddaljenost od središča v številu slikovnih točk. Enačba se ločeno uporablja v vertikalni in horizontalni smeri. Da dobimo končno vrednost v določeni točki, seštejemo vse prispevke za tisto točko.



Slika 3.6: Razdelitev slike velikosti 128×128 slikovnih točk, na 7×7 enakih delov velikosti 32 slikovnih točk, ki se med seboj v vseh smereh prekrivajo za 16 slikovnih točk. Rdeč in zelen kvadrat prikazujeta prvi dve podsliki.

S predlaganim pristopom pričakujemo pohitritev učenja, saj menimo, da je že del slike dovolj, da se mreža nauči celotne teksture objekta. Na primer, da se naučimo barvati vodo, ne potrebujemo celotnega območja vode na sliki, ki včasih lahko zavzema tudi pol ali več slike, ampak le ujemajoči del.

V tabeli 3.1 so prikazani časi potrebni za učenje pri dveh regresijskih

pristopih na 2.854.912 slikah, ki se razlikujeta le v tem, da en uči po delih, drugi pa na celih slikah. Oba pristopa sta bila učena ob istem času na isti napravi in ločenih GPU enotah. Iz tabele je razvidno, da lokalni pristop potrebuje krajši čas, da se nauči barvanja. To pripišemo predvsem krajšemu času potrebnemu za en korak učenja, število korakov do naučenega pristopa pa je podobno.

Tabela 3.1: Povprečen čas enega koraka (prehod preko 50.000 slik) pri učenju pristopov, število prehodov potrebnih za učenje in skupni čas potreben za učenje. Rezultati so prikazani za dva regresijska pristopa z enakimi arhitekturami, oba imata arhitekturo D+G in drugačnim načinom učenja. Iz rezultatov je razvidno, da lokalni pristopi za konvergenco potrebujejo manj časa.

Pristop	Čas za korak [s]	Št. korakov	Skupni čas učenja [s]
Lokalni pristop	619,8	90	55.782
Globalni pristop	1815,0	89	161.535

V tabeli 3.2 so predstavljene podrobnosti vsakega od lokalnih pristopov. Vsem pristopom je skupno, da uporabljajo cenilno funkcijo povprečna kvadratna napaka (MSE).

3.2.2 Globalni pristopi

Za primerjavo točnosti lokalnih pristopov s tistimi na celih slikah - globalnimi, smo dva pristopa, opisana v poglavju 3.2.1, pretvorili v globalni pristop. Uporabili smo arhitekturi D+G in D, ki smo jih prilagodili tako, da na vhodu sprejmeta celotno sivinsko sliko in vrneta celotno obarvano sliko. Tej smo dodali še globalni regresijski pristop z mrežo VGG, saj ta zaradi večkratnega pomanjšanja prostorskih dimenzij znotraj arhitekture ne more biti realiziran kot lokalni pristop. V tabeli 3.2 so predstavljeni vsi regresijski pristopi in njihove arhitekture. Pri vseh je cenilna funkcija povprečna kvadratna napaka (MSE).

Tabela 3.2: Regresijski pristopi in njihove arhitekture, ki so podrobneje opisane v poglavju 3.1.

Pristop	Arhitektura
Reg. lokalni	D+G
- brez softmax	D+G
- brez globalne mreže	D
Reg. globalni	D+G.
- brez globalne mreže	D
Reg. globalni VGG	X-VGG

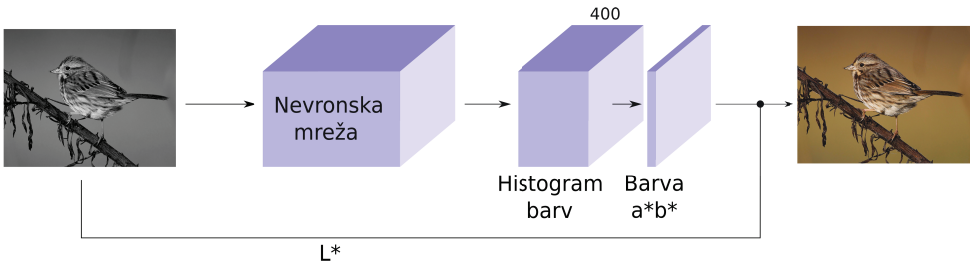
3.3 Pristopi s klasifikacijo

Razvili smo štiri pristope, ki namesto regresije uporabljajo klasifikacijo. Pri teh pristopih barvo ocenimo z izbiro najbolj primernega razreda, ki predstavlja nekaj sosednjih odtenkov. Posamezen razred opiše barve, ki so blizu skupaj v dvodimenzionalnem prostoru a*b*. Klasifikacija se izvede z uporabo *softmax* funkcije v zadnjem nivoju mreže.

Kot je prikazano na sliki 3.7, je vhod v metodo črno-bela slika, ki se posreduje nevronski mreži. Ta oceni obarvanje kot vektor verjetnosti za vsakega od razredov vsake slikovne točke. Te vrednosti se potem pretvorijo v komponenti $a*$ in $b*$ barvnega prostora CIE L*a*b*. Enako kot pri regresiji je rezultat združitev sivinske slike z ocenjenimi barvnimi komponentami.

Razrede smo dobili tako, da komponenti $a*$ in $b*$ barvnega prostora CIE L*a*b* razdelimo v 400 razredov, kar se je izkazalo za najbolje, saj s 400 razredi lahko barve zapišemo tako, da izgub zaradi nenatančnega kodiranja uporabnik ne bo opazil, večje število razredov pa bi preveč upočasnilo učenje in napovedovanje. Vsako od komponent smo razdelili v 20 razredov med vrednostima -100 in 100, kar pomeni, da vsak razred zajema interval širine 10. Vse kombinacije obeh komponent prinesejo 400 razredov.

Pretvorba iz zapisa $a*b*$ v histogram se uporabi pri učenju, kjer originalno sliko pretovorimo v histogram, da lahko izračunamo napako. To izvedemo z



Slika 3.7: Shematski prikaz delovanja klasifikacijskih pristopov, ki na vhodu prejmejo črno-belo sliko, s pomočjo nevronske mreže izračunajo verjetnosti za posamezen razred barv (histogram), te potem pretvorijo v barvni komponenti $a*$ in $b*$ barvnega prostora CIE L*a*b*, te pa nato združijo s sivinsko sliko $L*$, da dobijo obarvano sliko.

enačbo

$$l = 20 \left\lfloor \frac{a + 100}{10} \right\rfloor + \left\lfloor \frac{b + 100}{10} \right\rfloor, \tag{3.2}$$

kjer a in b predstavljata vrednost $a*$ in $b*$ slikovne točke, l pa indeks razreda v histogramu, ki zavzema cela števila v intervalu $[0, 399]$.

Pretvorba iz histograma v $a*$ in $b*$ se uporabi pri pretvorbi ocenjenih barvnih vrednosti s strani mreže in se izvede skladno z enačbo

$$\begin{aligned} a &= 10 \left\lfloor \frac{l}{20} \right\rfloor - 100 + 5 \\ b &= 10(l \bmod 20) - 100 + 5. \end{aligned} \tag{3.3}$$

Pri tem a in b predstavljata $a*$ in $b*$ barvne vrednosti slikovne točke, l pa predstavlja indeks razreda v histogramu, ki je bil napovedan z največjo verjetnostjo. Vsem komponentam dodamo še vrednost 5, tako da dobimo barvne vrednosti iz sredine vsakega razreda.

V tabeli 3.3 so prikazani pristopi s klasifikacijo, njihova arhitektura in cenilna funkcija. Vsi pristopi s klasifikacijo uporabljajo lokalni pristop.

Tabela 3.3: Lokalni klasifikacijski pristopi, njihove arhitekture, ki so podrobneje opisane v poglavju 3.1 in cenilne funkcije uporabljene za učenje. Cenilne funkcije so podrobneje opisane v poglavju 2.2.4

Pristop	Arhitektura	Cenilna funkcija
Klas. brez uteži - S+G.	S+G	Divergenca KL
Klas. brez uteži - D+G	D+G	Križna entropija
Klas. z utežmi - S+G	S+G	Križna entropija z utežmi
Klas. z utežmi - D+G	D+G	Križna entropija z utežmi

Poglavje 4

Vrednotenje

V nalogi smo pristope naučili na različno velikih učnih množicah, jih preizkusili na testni množici in rezultate vrednotili s primerjanjem z originalno sliko, za kar smo uporabili dve metriki. Ker ta primerjava v vseh primerih ne zadošča, smo pristope vrednotili še z anketo.

4.1 Postopek učenja

V tem poglavju bomo predstavili podrobnosti učenja na manjši množici, na večji učni množici in si za konec pogledali, kakšne značilke prepozna posamezen nivo nevronske mreže.

Za posodabljanje uteži mreže smo uporabili optimizator Adam, ki je trenutno najbolj v uporabi na nivoju nevronskih mrež. Pri tem so se za dobre izkazali parametri, ki so prikazani v tabeli 4.1. Prikazani parametri so bili izbrani z opazovanjem konvergence napake mreže, kjer so se izkazali za najboljšo izbiro. Pri vseh pristopih smo uporabili velikost serije (ang. *batch size*) 32, razen pri učenju metode Zhang in sod., kjer smo morali zaradi večjih dimenzij tenzorjev in posledično pomanjkanja pomnilnika, uporabiti velikost serije 8.

Za učenje na manjši učni množici smo izbrali vse pristope, za učenje na večji učni množici smo izbrali sedem pristopov, pet lastnih in dva iz sorodnih

Tabela 4.1: Parametri, s katerim smo Adam optimizatior nastavili.

Parameter	Vrednost parametra
stopnja učenja (ang. *learning rate*)	10^{-4}
beta 1	$0,9$
beta 2	$0,99$
epsilon	10^{-8}

del. Manjša in večja učna množica sta podrobno opisani v poglavju 4.3. Odločili smo se za tri regresijske pristope: lokalna regresija z globalno mrežo, globalna regresija z globalno mrežo in globalna regresija z mrežo VGG-16. Izpustili smo oba pristopa brez globalne mreže, saj imata očitno slabše rezultate barvanja. Izbrali smo dva klasifikacijska pristopa — klasifikacija brez uteži - S+G in klasifikacija z utežmi - S+G, ker se je izkazalo, da arhitektura D+G poslabša rezultate, zato smo se odločili, da izvedemo le primerjavo dveh pristopov z arhitekturo S+G.

4.2 Barvanje večjih slik

Večina pristopov v sorodnih delih je naučenih za barvanje slik velikosti 224×224 in ima to omejitev, da omogoča le barvanje slik te velikosti. Iizuak in sod. [10] omogočajo barvanje večjih slik, tako da večjo sliko podajo enaki mreži na vhod, ki nima omejitve za velikosti vhodne slike, saj je sestavljena le iz konvolucijskih nivojev. Pri tem avtorji komentirajo, da mreža deluje najbolje na slikah velikosti 224×224.

Z lokalnimi pristopi, ki smo jih implementirali v okviru tega dela, ta problem rešujemo drugače. Slike katerekoli velikosti večjih od 32×32 razdelimo na dele velikosti 32×32 s prekrivanjem, jo obarvamo po delih in potem spet sestavimo po principu opisanem v poglavju 3.2.1.

Primerjavo kakovosti barvanja večjih slik smo izvedli tako, da smo pristopa Iizuka in sod. ter lokalno regresijo z globalno mrežo preizkusili na istih slikah pomanjšanih na velikost 224×224 slikovnih točk, kjer naj bi bilo

barvanje optimalno in velikost 896×896. Primerjava je izvedena na mrežah naučenih na manjši učni množici s 100.000 slikami.

4.3 Podatki

Podatke, ki smo jih uporabili za učenje in validacijo pristopov, smo pridobili iz podatkovne zbirke ImageNet [21], ki vsebuje približno 14 milijonov slik. Iz zbirke smo naključno izbrali množico podatkov in jih za namen učenja pretvorili v barvni prostor CIE L*a*b*. Pri preizkusu na manjši množici smo za učenje naključno izbrali 100.000 slik, za validacijo pa 10.000 slik iz nabora. Validacijska množica je bila obenem tudi testna množica. Zaradi praktično neomejene količine podatkov smo se odločili za fiksno testno množico namesto prečnega preverjanja. Za učenje na večji množici smo naključno izbrali 2.854.912 slik.

Za vrednotenje barvanja večjih slik, opisanega v poglavju 4.2, podatki iz podatkovne zbirke ImageNet niso bili zadovoljivi, saj so slike večinoma velikosti manjših od 500×500 slikovnih točk. Odločili smo se, da testiranje izvedemo na slikah iz zbirke avtorja te naloge, ki so večje od velikosti 896×896, kar pomeni, da slik ni potrebno povečevati in s tem povzročati dodatnih napak v barvanju zaradi slabe kvalitete slik. Vzeli smo 584 slik, za katere je aplikacija Google Photos[1] ocenila, da gre za slike pohodništva. Za te slike smo se odločili, ker gre večinoma za slike narave, kjer je barvanje ponavadi najboljše in lahko razliko opazujemo na slikah, ki se večinoma barvajo dobro.

Za preizkus na starih črno-belih slikah smo izbrali raznovrstne slike iz dveh člankov iz spleta in zbirke starih črno-belih slik: 40 Must-See Photos From The Past[2], 37 Wonderfully Weird Old Photos That Show Just How Much We've Changed[3] in Old Photo Archive[4].

[1] http://photos.google.com
[2] http://www.boredpanda.com/must-see-historic-moments/
[3] http://www.lifebuzz.com/old-photos/
[4] http://oldphotoarchive.com/

4.4 Računanje napake

Za primerjavo pristopov smo napake računali na testni množici. Pri tem smo uporabili dve metriki: koren povprečne kvadratne napake (ang. *root mean squeared error*) ter razmerje med signalom in šumom (ang. *peak signal-to-noise ratio*).

Koren povprečne kvadratne napake (RMSE) za vsako sliko smo izračunali s pomočjo enačbe

$$\text{RMSE} = \sqrt{\sum_{i=1}^{h}\sum_{j=1}^{w}\sum_{k=1}^{c}(Y_{i,j,k} - \hat{Y}_{i,j,k})}, \tag{4.1}$$

kjer h in w prestavljata višino in širino slike ter c predstavlja število kanalov slike. Oznaka Y predstavlja originalno sliko (ang. *ground truth*) in $\hat{Y}$ obarvano sliko s strani pristopa za barvanje. Napaka je bila izračunana za vsako sliko posebej in kasneje povprečena preko vseh slik. Napaka RMSE je bila izračunana na slikah v barvnem prostoru CIE L*a*b* le za barvna kanala $a*$ in $b*$, saj za barvni kanal $L*$ izračun napake ni smiseln, ker so to vrednosti iz sivinske slike.

Razmerje med signalom in šumom (PSNR) je metrika, ki kaže razmerje med največjo možno močjo signala in močjo šuma, ki signal pokvari. Primerna je za primerjave rekonstruiranih podatkov, kot so v našem primeru validacijske slike, ki jih poskušamo rekonstruirati s pomočjo pristopov, ki bazirajo na nevronskih mrežah. Vrednosti razmerja med signalom in šumom se merijo v enoti decibel (dB) [31]. Zadovoljive vrednosti rekonstrukcije slike 8 bitnih podatkov, kamor sodijo tudi naši podatki, saj smo primerjali slike v barvnem prostoru *RGB*, so med 30 in 50 dB [32].

Napako PSNR smo izračunali z enačbo

$$\text{PSNR} = 20\log_{10}\left(\frac{MAX_I}{\text{RMSE}}\right), \tag{4.2}$$

v kateri ima MAX_I največjo možno vrednost slikovne točke, kar je v barvnem

prostoru RGB 255, RMSE pa je napaka izračunana z enačbo 4.1. Napako smo povprečili preko vseh slik v testni množici. Za izračun napake smo izbrali barvni prostor RGB, saj računanje PSNR v prostoru L*a*b* ni možno, ker ne poznamo največje možne vrednosti signala za komponenti $a*$ in $b*$.

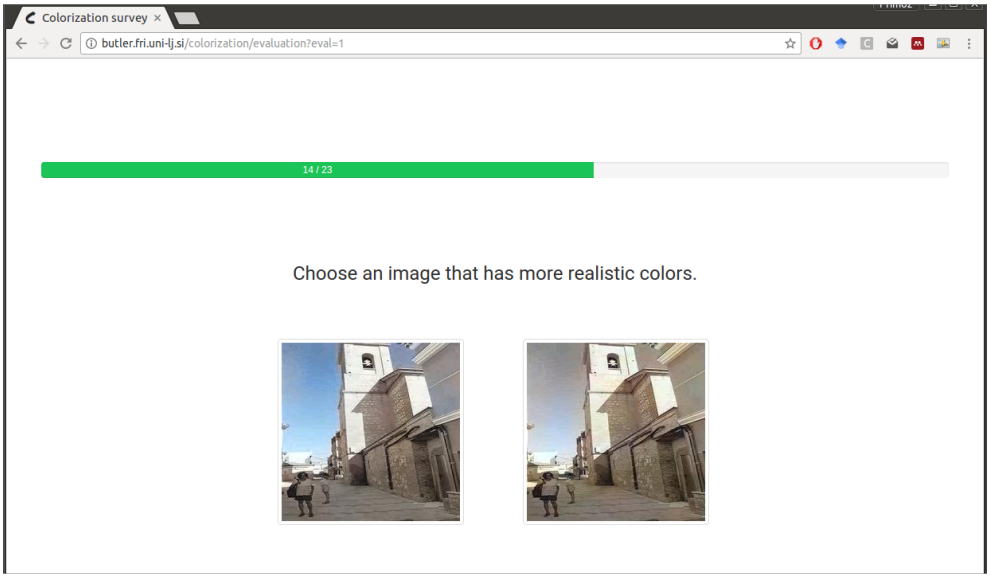
4.5 Primerjava pristopov glede na realističnost barvanja

Izkaže se, da številčna napaka, izračunana s primerjavo z originalno sliko, ni vedno dobra metrika za kvaliteto barvanja, saj nas bolj kot natančna obarvanost zanima realističnost in naravnost slike. Številčna napaka ni najboljša metrika predvsem v naslednjih primerih:

- Nekateri objekti na sliki imajo lahko različne barve, zato v primeru, ko algoritem pobarva z drugačno, a še vedno smiselno barvo, primerjava z originalno sliko ne da pravilne ocene napake.
- Odtenki na originalni sliki so lahko različno močni, medtem ko barvanje lahko izgleda enako naravno pri različno močnih barvah. Na primer travnik je lahko obarvan z bolj nežnimi sepia barvami ali bolj močno zeleno barvo. Oboje izgleda naravno, medtem ko številčna napaka preferira eno od teh rešitev.

Zaradi omenjenih problemov s številčno napako smo se odločili, da pristope primerjamo tudi s pomočjo spletne ankete. Anketirancem smo na zaslonu pokazali eno sliko obarvano z dvema različnima pristopoma, ti pa so morali oceniti, katera slika je bolje, oziroma bolj realistično obarvana. Slika 4.1 prikazuje aplikacijo za ocenjevanje. Vsak anketiranec je ocenil 23 parov slik, od katerih sta bili dve testni in nista upoštevani v evalvaciji. Ostalih 21 je izbranih tako, da je vsak od anketirancev vsako od kombinacij pristopov ocenil enkrat. Anketiranec je vedno dobil drugo sliko, s čimer smo zmanjšali vpliv že videne slike na oceno. Slike smo izbirali po principu naključne zasnove poizkusa (ang. *randomized experimental design*) [33]. Za tako zasnovo

smo se odločili, da izničimo vse vplive, ki bi jih lahko povzročil določen vrstni red slik.



Slika 4.1: Aplikacija za vrednotenje kakovosti barvanja slik s pomočjo anketirancev. Anketiranec je v posameznem koraku dobil na zaslon sliko obarvano z dvema pristopoma. S klikom na sliko je ocenil, katera je bolj naravno obarvana.

Evalvacijo smo izvajali na skupno 100 slikah, ki so bile izbrane naključno iz množice 10.000 testnih slik. V evalvacijo je bilo vključenih sedem pristopov naučenih na večji učni množici, ki so našteti v poglavju 4.1.

Spletno anketo za ocenjevanje smo implementirali v ogrodju Django[5] in je vsebovala štiri prikaze. Prvi je bil zaslon s kratkimi navodili za uporabnika. Sledil je zaslon, ki je bil namenjen kratkemu opisu poteka evalvacije, s čimer smo poskrbeli, da uporabnik na prvem zaslonu ni dobil preveč informacij. Sledilo je 23 ponovitev tretjega zaslona, na katerem je potekala evalvacija in je prikazan na sliki 4.1. Na koncu je sledil zaslon z zahvalo in povabilom na ponovno evalvacijo. V primeru, da se je uporabnik odločil za ponovno evalvacijo, smo poskrbeli, da je vedno dobil nove slike, ki jih

[5] `https://djangoproject.com`

prej še ni videl. V primeru, da je ocenil že vse slike, mu je bila nadaljnja evalvacija onemogočena. Izvorna koda aplikacije je objavljena na GitHubu PrimozGodec/EvaluationApp[6].

Anketiranje je potekalo v dneh od 25. do vključno 27. julija 2017. Anketirance smo pridobili z objavo na socialnih omrežjih Facebook in LinkedIn. V anketi smo zbrali odzive 1010 anketirancev. Anketiranci so v povprečju ocenili $21,49$ slik, če iz statistke izključimo dve testni sliki, ki sta bili namenjeni privajanju na način evalvacije. Večina anketirancev, 838, je izpolnila anketo samo enkrat, 90 anketirancev se je odločilo za vsaj eno ponovno evalvacijo, 82 anketirancev pa je anketo zapustilo predčasno in so ovrednotili manj kot 21 slik. Zbrani podatki v obliki datoteke *json* so objavljeni na GitHubu PrimozGodec/EvaluationApp[7].

[6] https://github.com/PrimozGodec/EvaluationApp
[7] https://github.com/PrimozGodec/EvaluationApp

Poglavje 5

Rezultati in diskusija

V tem poglavju predstavljamo natančnosti pristopov naučenih na manjši množici in jih med seboj primerjamo. Pogledali si bomo primerjavo pristopov naučenih na večji učni množici in primerjali pristope pri barvanju večjih slik od tistih, na katerih so bili pristopi naučeni.

5.1 Primerjava pristopov na manjši učni množici

Tabela 5.1 prikazuje natančnost pristopov na testni množici slik. Izkaže se, da se na manjši množici najbolje obnese pristop Iizuka in sod., ki je glede na napako RMSE za $0,066$ boljši od našega pristopa z globalno regresijo. Ostali pristopi, razviti s strani drugih avtorjev, se na tej množici obnesejo slabše od večine naših pristopov.

Izkazalo se je tudi, da se na tej množici, glede na napako, regresijski pristopi obnesejo bolje kot pristopi s klasifikacijo. Pristop lokalna regresija se s klasifikacijo brez uteži z arhitekturo D+G glede na RMSE razlikuje skoraj za 2. Omenjena pristopa imata enako arhitekturo mreže. Med pristopi z regresijo je opaziti boljše rezultate pri tistih, ki barvajo celotno sliko naenkrat. Opazimo lahko tudi, da globalna mreža prinese izboljšave glede na RMSE nekje od $0,3$ do $0,4$. Pri klasifikacijskih pristopih se je izkazalo, da arhitek-

tura S+G deluje bolje pri tej učni množici. Izkaže se tudi, da uteži v cenilni funkciji ne prinesejo izboljšave v natančnosti.

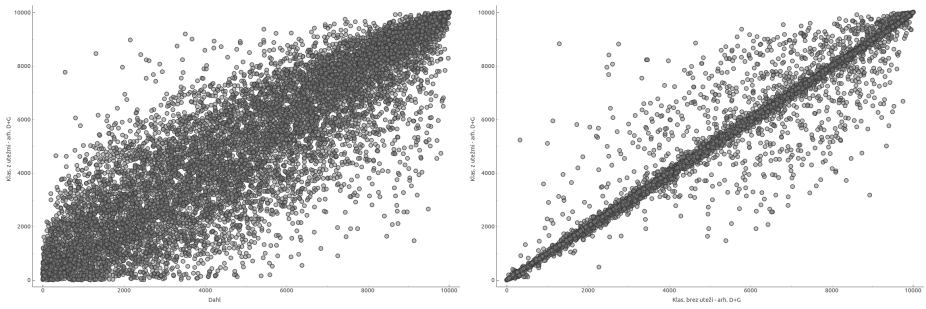
Tabela 5.1: Tabela prikazuje napake izračunane na testni množici podatkov. Za vsakega od pristopov smo izračunali napaki opisani v poglavju 4.4. V zgornjem delu tabele so prikazane napake na pristopih iz sorodnih del, v vmesnem napake na pristopih z regresijo in v spodnjem delu tabele napake na pristopih s klasifikacijo.

Pristop	RMSE	PSNR
Zhang in sod.	15,004	22,252
Iizuka in sod.	12,941	23,439
Dahl	13,936	22,551
Reg. lokalna	13,216	23,199
- brez softmax	13,206	23,183
- brez globalne mreže	13,767	22,840
Reg. globalna	13,007	23,434
- brez globalne mreže	13,334	23,068
Reg. globalna VGG	13,387	23,131
Klas. brez uteži - S+G	14,336	22,738
Klas. brez uteži - D+G	15,086	22,380
Klas. z utežmi - S+G	14,573	22,610
Klas. z utežmi - D+G	15,137	22,395

Za vsak pristop smo napake za slike iz testne zbirke spremenili v range glede na napako RMSE na določeni sliki, ki se raztezajo od najboljše z rangom 0, do najslabše z rangom 9.999. Izkaže se, da rangi med pristopi močno korelirajo (slika 5.1). To pomeni, da je natančnost barvanja slike v veliki meri odvisna od motiva na sliki, kar je bilo pričakovati.

Korelacija je v veliki meri prisotna pri vseh pristopih, je pa nekoliko različna glede na sorodnost pristopov (tabela A.1 v prilogi). Pristopa klasifikacija brez uteži - D+G in klasifikacija z utežmi - D+G (slika 5.1, desno), ki sta si bolj podobna glede na arhitekturo in način barvanja, imata tako zelo veliko korelacijo. Pristopa klasifikacija brez uteži - D+G in Dahl (slika 5.1, levo), ki sta si na način napovedovanja bolj različna, prvi je regresijski in drugi klasifikacijski, imata manjšo korelacijo. Še vedno so točke razpore-

jene okoli premice, ki razpolavlja kvadrant grafa, vendar je odstopanj več. Podobne slike dobimo tudi pri primerjavi ostalih pristopov.



Slika 5.1: Primerjava rangiranja slik glede na napako RMSE pri dveh različnih pristopih barvanja. Os X predstavlja rang pri prvem pristopu, Y pa rang pri drugem pristopu. Prva slika prikazuje primerjave rangov Dahlovega pristopa in klasifikacijskega pristopa z arhitekturo D+G. Druga slika prikazuje range pri dveh klasifikacijskih pristopih z enakimi arhitekturami.

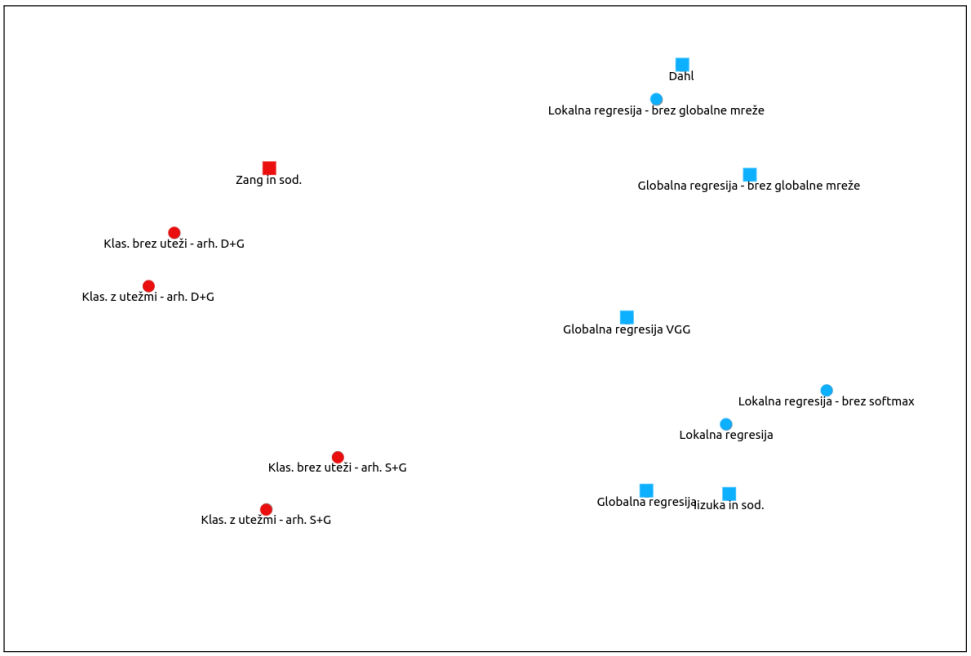
Ker smo želeli podobnost pristopov med seboj primerjati v prostoru, smo izračunali Spearmanovo korelacijo rangov [34] za vsak par pristopov. Korelacije so številčno prikazane v tabeli A.1 v prilogi. Za izris podobnosti (slika 5.2) smo uporabili metodo večdimenzionalno skaliranje (ang. *multidimensional scaling* - MDS) [35] na Spearmanovih korelacijah. Uporabili smo implementacijo v programu Orange [36].

Pristopi s klasifikacijo so skupaj (levo na sliki) in so bolj oddaljeni od tistih z regresijo (desno na sliki). Pri pristopih s klasifikacijo opazimo, da na različnost bolj vpliva vrsta arhitekture kot uporaba uteži za pogostost barve. Izkaže se, da je pristop Zhang in sod. bližje tistim z arhitekturo D+G. Glede na to, da so lastnosti teh arhitektur popolnoma drugačne, se izkaže, da na podobnost glede na napake vpliva predvsem globina arhitekture.

Pri regresijskih pristopih lahko opazimo, da je največja razlika glede na uporabo globalne mreže. Pristopi, ki ne uporabljajo globalne mreže, so na vrhu prikaza, ostali so spodaj. Globalna regresija VGG je bližja pristopom z globalno mrežo. To gre verjetno pripisati dejstvu, da ta pristop uporablja mrežo VGG-16, ki je del globalne mreže pri ostalih pristopih, vendar ta

pristop to mrežo izkorišča kot glavno.

Čeprav je Dahlov pristop glede na arhitekturo popolnoma različen od naših, se glede na napake izkaže podoben pristopom brez globalne mreže, kar še dodatno potrdi deljenje pristopov glede na prisotnost globalne mreže. Pristop Iizuka in sod. je v gruči s pristopi, ki uporabljajo globalno mrežo, saj tudi sam uporablja podoben pristop. Pri pristopih z regresijo lahko opazimo še, da sta pristopa Iizuka in sod. ter globalni regresijski pristop bolj skupaj, saj oba delujeta na celotni sliki.

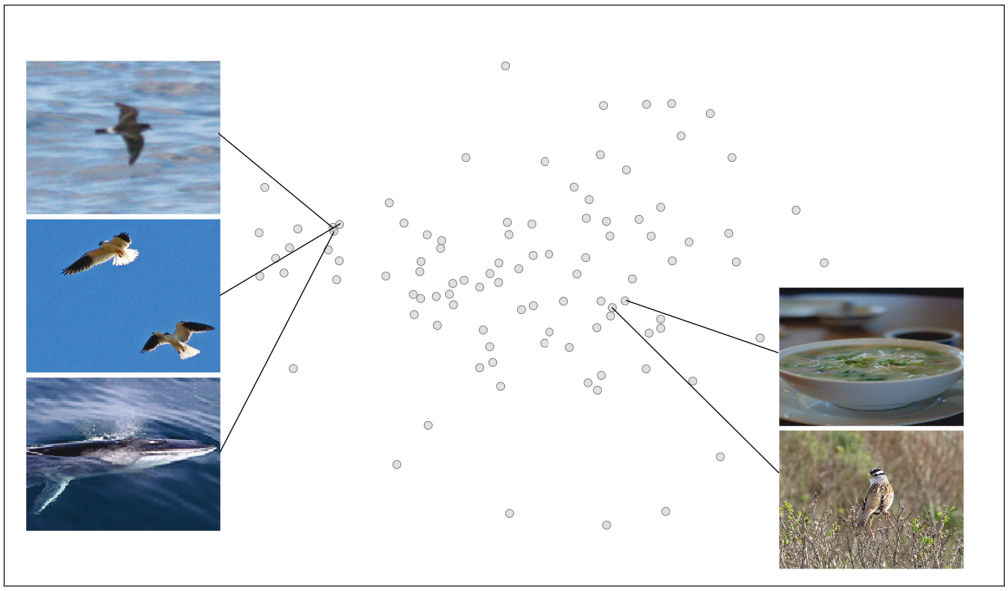


Slika 5.2: Primerjava pristopov v prostoru MDS kaže na sorodnosti med pristopi glede na vrsto pristopa (klasifikacija - označeno z rdečo in regresija - označena z modro), arhitekturo mreže in načinom napovedovanja (lokalni - označeno s krogom ali globalni - označeno s kvadratom).

Zanimalo nas je tudi, katere so tiste slike, kjer je eden od pristopov boljši, ostali pa slabši. Na sliki 5.1 so taki primeri točke, ki ležijo stran od diagonale. Te slike smo našli z metodo za iskanje osamelcev (ang. *outliers*) [37], ki poišče slike, ki so najbolj oddaljene od diagonale v večdimenzionalnem prostoru vseh pristopov.

Slike, ki najbolj izstopajo glede na napako na različnih pristopih, so prikazane kot točke v prostoru MDS, konstruiranem glede na range slik (slika 5.3). Ob pregledu slik se izkaže, da gre tukaj večinoma za slike, kjer je težje zaznati teksturo. Ob tem predvidevamo, da so se določeni pristopi bolje naučili ravno te teksture kot drugi pristopi.

Napaka je močno povezana z motivom na sliki. V prostoru MDS so bližje skupaj slike s podobnim motivom in barvami. Na levi strani so prikazane tri slike, ki so blizu skupaj in jim je skupno to, da je na sliki morje ali nebo, ki sta oba modre barve in imata prisoten določen objekt (v našem primeru žival). Na desni strani sta dve sliki, ki se tudi ujemata glede na odtenke v sliki, čeprav je motiv popolnoma drugačen.



Slika 5.3: Razporeditev slik v prostoru MDS, ki zajema 100 slik, ki najbolj izstopajo glede na različnost rangov pri pristopih. Prostor je konstruiran glede na kosinusno razdaljo med vektorji rangov slik. Opazimo lahko, da so v prostoru podobne slike bližje skupaj. Dve podobni skupini slik sta prikazani ob robu.

Primerjave barvanja slik po pristopih so prikazane na sliki 5.4 za regresijo in na sliki 5.5 za klasifikacijo. Slike smo izbrali tako, da prva dva stolpca prikazujeta dve sliki iz množice dvajset najbolje obarvanih s strani vseh pri-

stopov, tretji in četrti stolpec prikazujeta slike, ki so bile različno dobro obarvane s strani različnih pristopov. Ti sliki sta izbrani izmed točk v prostoru na sliki 5.3. Zadnja dva stolpca prikazujeta slike, ki so bile v množici dvajset najslabše obarvanih s strani vseh algoritmov.

Opazimo lahko, da sliki v prvih dveh stolpcih spadata v skupino najbolje obarvanih slik zato, ker imajo že originalne slike prisotne zelo nenasičene (blede) odtenke barv. Pristopi, posebej regresijski, ki običajno obarvajo z bolj nenasičenimi barvami, so se zato zelo približali originalni sliki, čeprav barvanje v več primerih ni ravno najboljše. Pri drugem in tretjem stolpcu so se nekateri pristopi dobro približali pravi barvi, drugi pa so slike obarvali napačno. Sliki iz zadnjih dveh stolpcev sta bili glede na napako v množici najslabše obarvanih slik zato, ker imajo originalne slike zelo močne odtenke, katerim se pristopi niso približali. Kljub temu so nekatera barvanja dovolj naravna v primeru, da jih ne primerjamo z originalno sliko.

V primerjavi s pristopi iz sorodnih del tudi tu opazimo, da najboljše barva pristop Iizuka in sod., ki je imel tudi najmanjšo napako. Zhang in sod. se na določenih delih obnese dobro, vendar so slike zelo lisaste in nepopolno obarvane, medtem ko so pri pristopu Dahl odtenki zelo rjavi, čeprav vmes lahko opazimo nekaj pravih barv.

Pri primerjavi regresijskih pristopov lahko opazimo, da je po pričakovanjih najboljše barvanje s strani globalne regresije z globalno mrežo, čeprav lokalna regresija ne zaostaja dosti. Opazimo lahko tudi pomen in izboljšavo z uporabo globalne mreže. Enaki pristopi brez globalne mreže so obarvali bolj nenatančno, nenaravno, prisotnih je tudi več rjavih odtenkov.

Pri klasifikacijskih pristopih opazimo, da pristopi z arhitekturo S+G dajejo boljše rezultate kot tisti z D+G, kjer barvanja skorajda ni. Pri pregledu slik in primerjavi klasifikacije z utežmi in brez na slikah opazimo, da pristop z utežmi barva z močnejšimi odtenki kot tisti brez, kar je bilo za pričakovati, saj je namen uteži zmanjšati izbor šibkejših odtenkov, ki imajo $a*$ in $b*$ vrednost bližje nič. Kljub barvanju z močnejši odtenki barv in s tem približevanju realni barvi, je na pogled barvanje brez uteži bolj naravno, saj je

opaziti manj napak v barvanju. Za primer lahko vzamemo sliko s ptico, kjer sta les in ptica pobarvana bolj realno in letalo nima sivega pasu okoli sebe.

Iz primerjave barvanja na slikah 5.4 in 5.5 lahko opazimo, da pristopi z regresijo obarvajo bolje in bolj naravno kot pristopi s klasifikacijo, čeprav je nekaj izjem pri barvanju neba in praproti. Izkaže se tudi, da barvanje z regresijo večkrat obarva z bolj rjavimi odtenki, kar pri klasifikacijskih pristopih ni zaznati. Tam je bolj pogosto, da slika ni obarvana.

5.2 Primerjava pristopov na večji učni množici

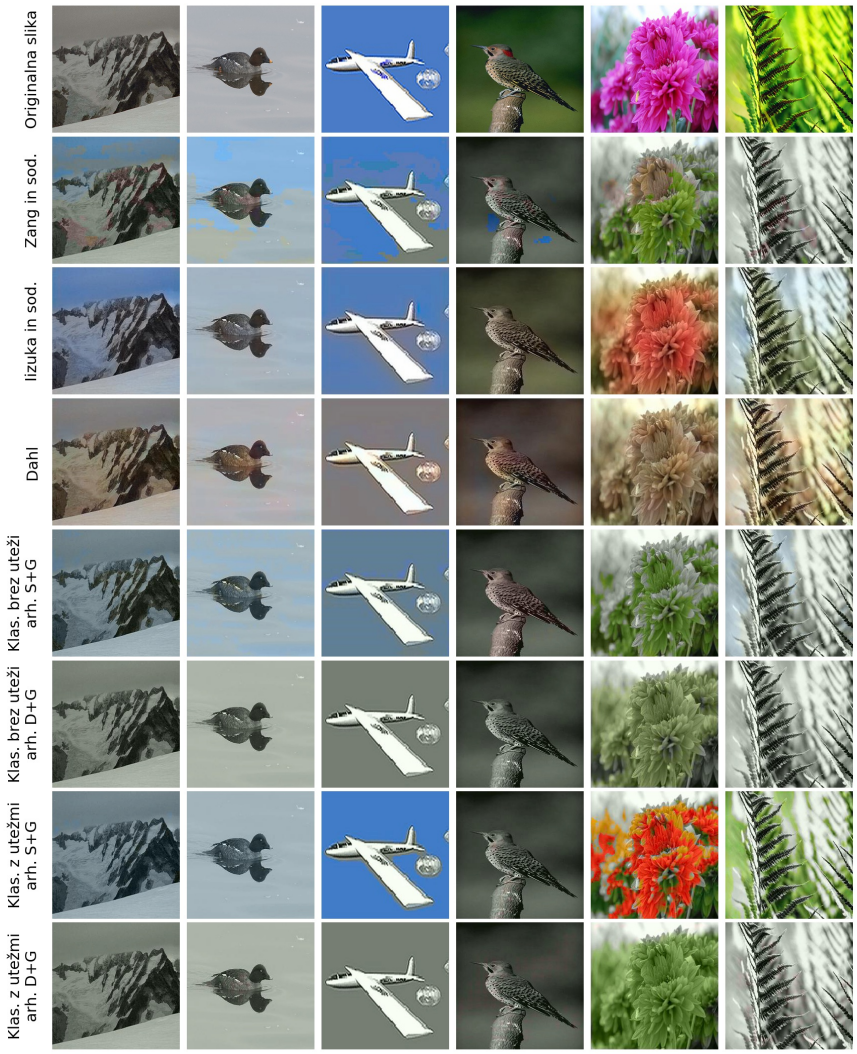
V tabeli 5.2 so prikazane napake pristopov, ki so naučeni na večji učni množici. Opazimo, da je razmerje med pristopi ostalo nespremenjeno glede na rezultate v poglavju 5.1. Prav pri vseh pristopih se je po pričakovanjih izboljšala natančnost barvanja. Kot lahko opazimo, če rezultate primerjamo s tistimi v tabeli 5.1, smo največjo izboljšavo pridobili pri pristopu Iizuka in sod. in pristopu klasifikacija z utežmi z arhitekturo S+G.

Tabela 5.2: Napake izbranih pristopov izračunane na testni množici podatkov. Za vsakega od pristopov smo izračunali dve napaki opisani v poglavju 4.4. V zgornjem delu tabele so prikazane napake na pristopih iz sorodnih del, v vmesnem napake na pristopih z regresijo in v spodnjem delu tabele napake na pristopih s klasifikacijo.

Pristop	RMSE	PSNR
Iizuka in sod.	12,252	23,831
Dahl	13,745	22,827
Reg. lokalna	12,960	23,363
Reg. globalna	12,368	23,829
Reg. globalna VGG	12,976	23,398
Klas. brez uteži - S+G	14,015	22,909
Klas. z utežmi - S+G	14,326	22,717

Slika 5.6 prikazuje izboljšanje barvanja po prvih desetih korakih učenja.

Slika 5.4: Slike iz testne množice, ki so bile obarvane z regresijskimi pristopi opisanimi v tem delu in pristopi iz sorodnih del. Vsaka vrstica prikazuje drug pristop, prva vrstica prikazuje originalno sliko. Sliki v prvih dveh stolpcih sta bili izbrani iz množice 20 najbolje obarvanih slik glede na rang, srednji dve iz množice 100 slik, ki najbolj izstopajo glede na različnost rangov pri pristopih in zadnji dve iz množice 20 najslabše obarvanih slik.



Slika 5.5: Slike iz testne množice, ki so bile obarvane s klasifikacijskimi pristopi opisanimi v tem delu in pristopi iz sorodnih del. Vsaka vrstica prikazuje drug pristop, prva vrstica prikazuje originalno sliko. Sliki v prvih dveh stolpcih sta bili izbrani iz množice 20 najbolje obarvanih slik glede na rang, srednji dve iz množice 100 slik, ki najbolj izstopajo glede na različnost rangov pri pristopih in zadnji dve iz množice 20 najslabše obarvanih slik.

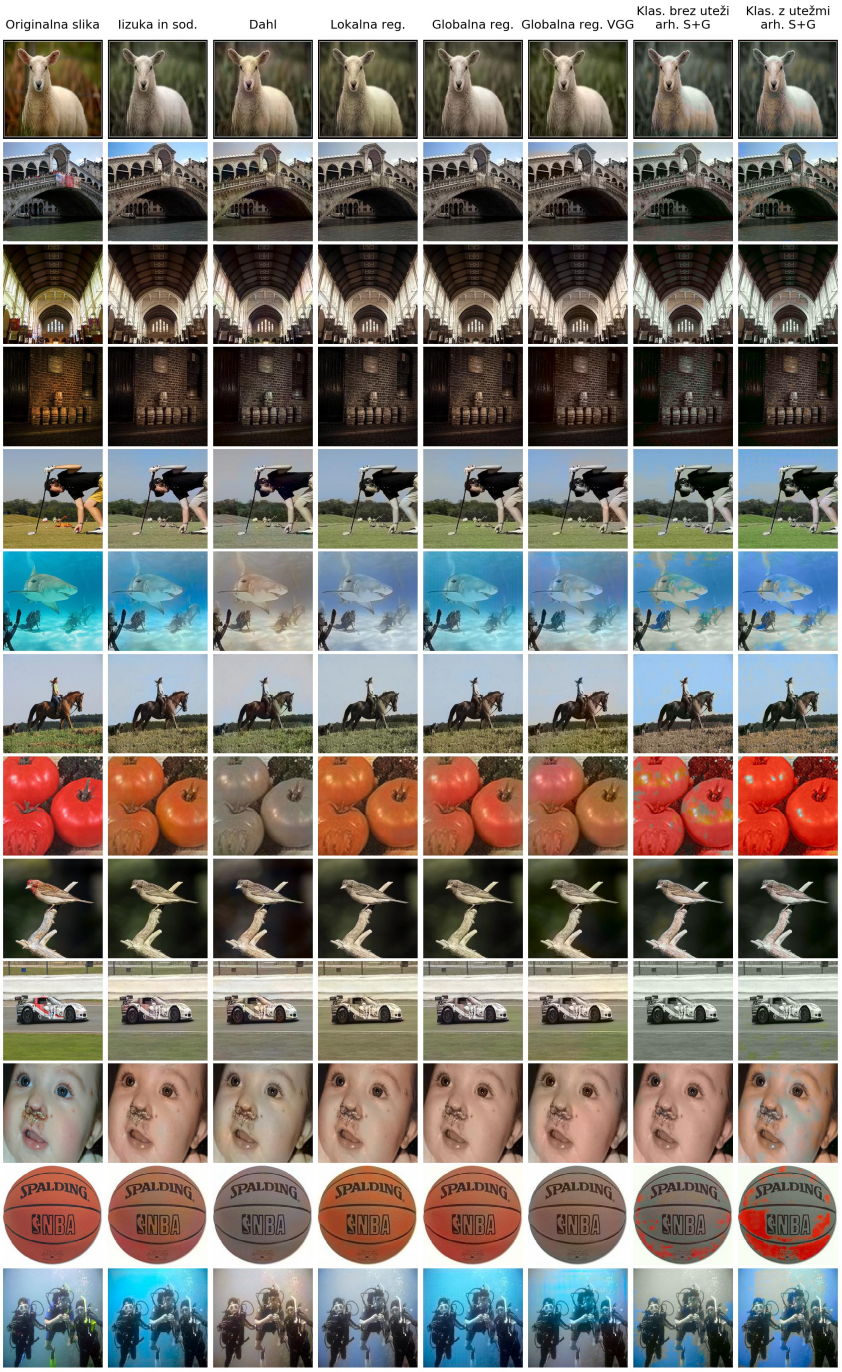
Korak predstavlja učenje na sklopu 50.000 slik. Deset korakov smo izbrali, ker je tu viden največji napredek. Izkaže se, da je pri vseh pristopih največja razlika narejena že v prvem koraku, v nadaljnjih pa se dogajajo spremembe, ki večinoma barvanje naredijo bolj realno.

Za konec predstavitve smo izbrali še nekaj slik, ki so bile dobro obarvane pri skoraj vseh pristopih in nekaj slik, katerih barvanje je bilo slabo ali delno slabo v vseh primerih.

Dobro obarvane slike so prikazane na sliki 5.7. Te slike so v večini slike narave in tiste z zelo pogostimi motivi, ki imajo enako barvo v vseh primerih. Na primer rdeč opečnat zid na zadnji sliki je zelo dobro obarvan, saj je vedno enake barve, prav tako rdeč paradižnik. Opazimo lahko, da pristopi Iizuka in sod ter lokalna regresija in globalna regresija v večji meri barvajo bolj natančno. Izkaže se, da pristopa Iizuka in sod. in globalna regresija nimata bistvenih razlik v kakovosti barvanja pri slikah, ki sodijo v množico dobro obarvanih slik. Sklepamo, da se globalni regresijski pristop izkaže za slabšega od pristopa Iizuka in sod. na slikah, ki sodijo v množico slabše obarvanih slik. Lokalna regresija nekoliko zaostaja, vendar je barvanje še vedno realno. Klasifikacijska pristopa se izkažeta za manj natančna, vidi pa se, da pristop z utežmi poskuša barvati z močnejšimi odtenki. To se najbolje opazi pri paradižniku, ki je še najbolje obarvan prav s tem pristopom. Izkaže se, da so največje razlike pri barvanju pri podvodnih slikah, mogoče bi to lahko pripisali manj očitnim teksturam v sliki. Presenečeni smo nad kvaliteto barvanja košarkarske žoge.

Slika 5.8 prikazuje slike, ki so obarvane slabše. Slike so obarvane slabo iz več razlogov. Pri večini gre za problem pri predmetih z neenoličnimi barvami. V tem primeru nekateri pristopi obarvajo naravno, vendar drugače kot je v originalu, s čimer ni nič narobe, nekateri pa obarvajo popolnoma narobe. Seveda je vse odvisno od motiva. Šopek rož je na primer praktično pri vseh regresijskih pristopih obarvan naravno. Tudi jakna na sliki je v nekaterih primerih obarvana naravno, medtem ko notranjost prostorov in zunanjost hiš ni prepričljivo obarvana. V nekaterih primerih pride do napake, ko določen

Slika 5.6: Napredovanje barvanja po prvih desetih korakih učenja pri različnih prestopih. Stolpec predstavlja pristop, vsaka vrstica pa posamezen korak.



Slika 5.7: Slike iz testne množice, katerih barvanje se je izkazalo za dobro. Vsak stolpec predstavlja enega od pristopov opisanih v delu.

predmet ali žival dobi barvo okolice. To se večkrat zgodi pri lokalnih pristopih vendar ni zelo običajno. Sklepamo lahko, da se ti pristopi občasno preveč opirajo na globalno mrežo. Veliko slik je obarvanih s premalo nasičenimi barvami. To se zgodi pri predmetih, ki nimajo enolične barve. Na primer rdeč avto kaže zametke rdečih odtenkov, ki pa niso ravno prepričljivi. Pri sliki v drugi vrsti so skalne stene zelo močno rdečih odtenkov. Ker mreža večkrat vidi primer sivih skal, je tudi tukaj barvanje bolj sivo kot rdeče.

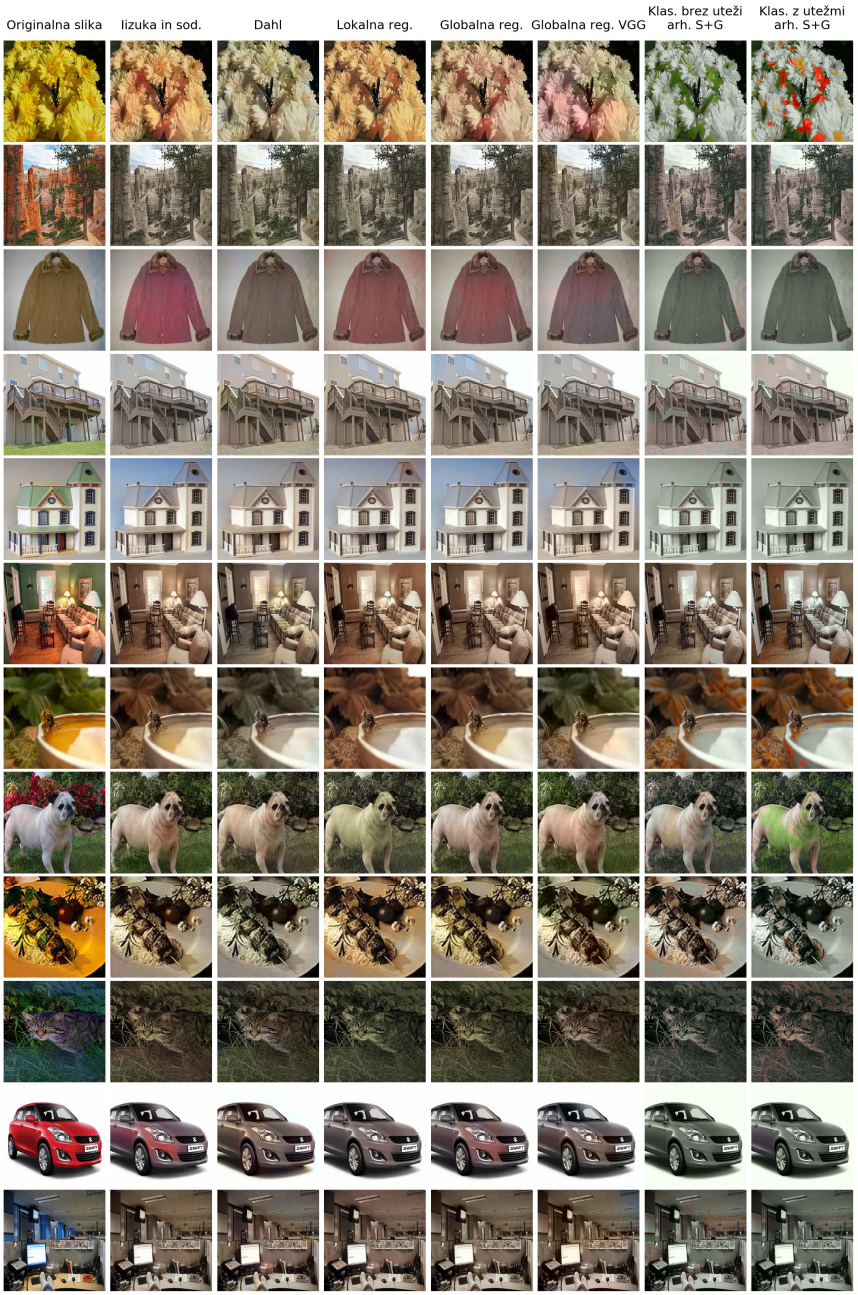
Kot smo ugotovili že v poglavju 5.1, lahko tudi na primerih slik zaključimo, da je kvaliteta barvanja v veliki meri odvisna od motiva na sliki. Večino dobro obarvanih slik je pri vseh pristopih obarvano pristopu primerno, medtem kot tiste slabo obarvane, večinoma slabo obarvane pri vseh pristopih. Barvanje se redko približa kvaliteti originalne slike, vendar v večini primerov uporabi prave barve, odtenki pa niso vedno najbolj prepričljivi.

5.3 Primerjava pristopov glede na realističnost barvanja

Za primerjavo pristopov smo zgradili model Bradley–Terry [38]. Parametre smo ocenili z metodo največjega verjetja (ang. *maximum likelihood*). Model je zgrajen tako, da nam je v pomoč pri rangiranju podatkov pridobljenih v medsebojni primerjavi parov. Pri katerem ni nujno, da so primerjave popolne in enako pogoste za pare.

Model je bil zgrajen na podlagi primerjav prikazanih v tabeli C.1 v prilogi. Izračunana sta bila parametra γ_i in β_i, ki sta si v razmerju $\gamma_i = e^{\beta_i}$, kjer je i zaporedna številka pristopa. Večja vrednost parametra pomeni pristop, ki se je bolje izkazal glede na evalvacijo. Parametri so prikazani v tabeli 5.3.

Če rezultate primerjamo s tistimi v poglavju 5.2, kjer smo pristope primerjali glede na izračunano napako, vidimo določene podobnosti in tudi razlike. Najbolje se je enako kot pri napaki s primerjavo z originalno sliko izkazal pristop Iizuka in sod. Zaostaja naš globalni regresijski pristop, do spremembe pa je prišlo pri globalnem regresijskem pristopu z mrežo VGG, ki se je glede



Slika 5.8: Slike iz testne množice, katerih barvanje se je izkazalo za slabše. Vsak stolpec predstavlja enega od pristopov opisanih v delu.

Tabela 5.3: Vrednosti parametrov modela Bradley-Terry za pristope, ki jih primerjamo. Večja vrednost pomeni, da se je pristop bolje izkazal pri ocenjevanju.

Pristop	γ_i	β_i
Iizuka in sod.	0,277	-1,28
Dahl	0,125	-2,08
Reg. lokalna	0,106	-2,25
Reg. globalna	0,189	-1,67
Reg. globalna VGG	0,130	-2,04
Klas. brez uteži - S+G	0,098	-2,32
Klas. z utežmi - S+G	0,076	-2,58

na oceno anketirancev izkazal za boljšega od lokalne regresije. Enako kot prej je pristop, ki ga je razvil Dalh, boljši od klasifikacijskih, ki se tudi v primeru evalvacije s pomočjo anketirancev izkažeta za slabša.

5.4 Barvanje večjih slik

Tabela 5.4 prikazuje napake pri dveh velikostih slik na dveh pristopih. Ena velikost je velikost na kateri je bila mreža naučena, druga pa je štirikratna velikost slik v višino in širino. Izkaže se, da ima pristop lokalna regresija z globalno mrežo zelo majhno razliko v napaki, pri barvanju slik večjih velikosti glede na napako na osnovni velikosti, medtem ko je ta razlika pri pristopu Iizuka in sod. merjena v RMSE kar $6,18$.

5.5 Barvanje starih slik

Glavna motivacija magistrskega dela je bila razviti pristop za barvanje črno-belih slik z namenom, da obarvamo stare črno-bele slike. Zato smo pristop uporabili na množici zgodovinskih slik. Za barvanje zgodovinskih slik smo uporabili pristop, ki se je najbolje obnesel, to je globalna regresija.

Tabela 5.4: Primerjava napak pristopov Iizuka in sod. ter lokalna regresija z globalno mrežo pri barvanju dveh velikosti slik. 224×224 je velikost na kateri je bila mreža naučena, druga velikost je uporabljena za testiranje razlike v barvanju večjih slik. Za računanje napake so uporabljene metrike opisane v poglavju 4.4.

Pristop	Velikost slik	RMSE	PSNR
Iizuka in sod.	224	10,018	24,750
	896	16,136	20,906
Reg. lokalna	224	9,892	24,700
	896	10,096	24,523

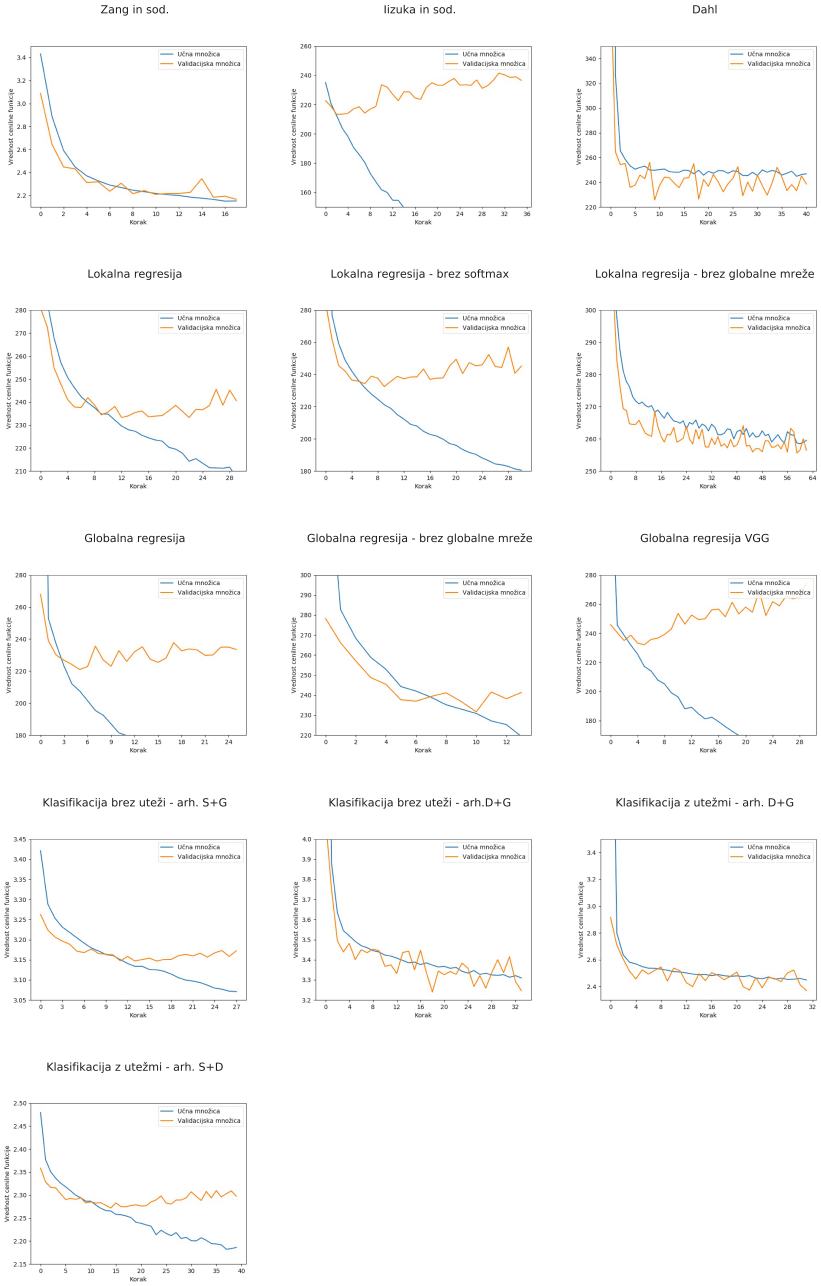
Rezultati barvanja zgodovinskih slik so prikazani na sliki 5.9. Iz rezultatov lahko opazimo dobro barvanje objektov z enolično določeno barvo. Nekaj več napak zaradi pomanjkljive kvalitete slik lahko opazimo na nekaterih delih. Tam, kjer je barva neenolična, je prisotnih več rjavih odtenkov, česar smo že vajeni iz prejšnjih primerov. Slike niso enake tistim, ki bi bile zajete z barvnim fotoaparatom, če bi obstajal, ampak so bolj nežnih in nenasičenih odtenkov. V večini primerov so barve smiselne in doprinesejo k živahnosti fotografije.

5.6 Konvergenca nevronske mreže

Pri učenju smo beležili vrednosti cenilne funkcije po vsakem končanem prehodu čez vse podatke (ang. *epoch*) na učni množici in validacijskih množici. S tem smo opazovali, kdaj je določen pristop optimalno naučen. To se v našem primeru zgodi v trenutku, ko vrednosti cenilne funkcije na validacijski množici prenehajo padati ali začnejo naraščati. V tem trenutku je naša mreža optimalno naučena, zato smo te uteži uporabili za testiranje. Grafi padanja cenilnih funkcij za vse pristope naučene na manjši učni množici so prikazani na sliki 5.10.

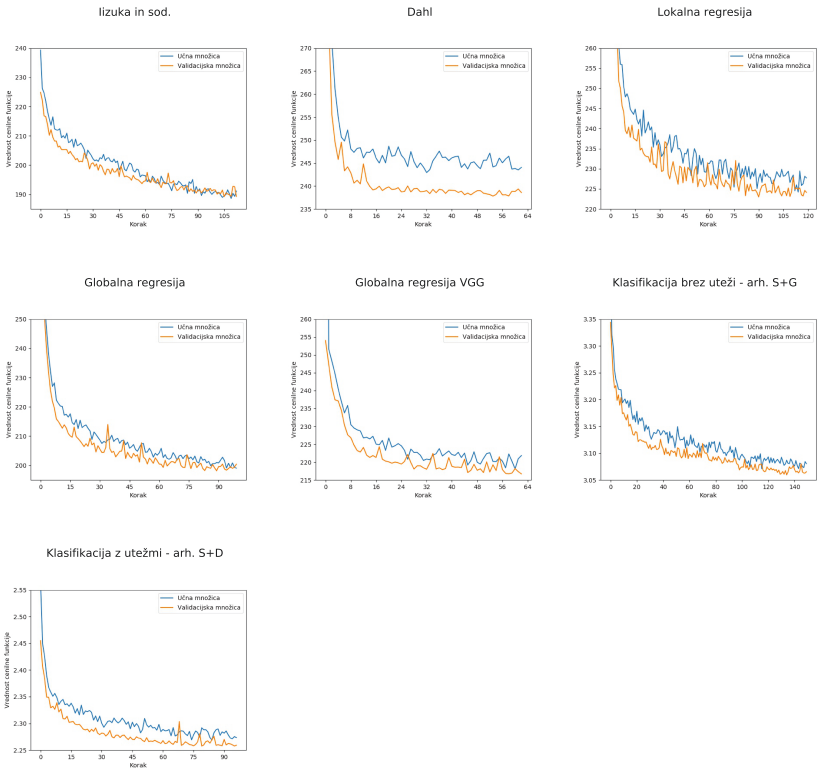
Enako kot v primeru na manjši učni množici smo si tudi pri večji izrisovali

Slika 5.9: Pari črno-belih zgodovinskih slik in obarvanih slik. Slike so pridobljene iz spletnih podatkovnih zbirk navedenih v poglavju 4.3.



Slika 5.10: Prikaz padanja napake modela pri učenju na manjši množici. Za vsak prehod preko vseh podatkov (ang. *epoch*) je prikazana vrednost cenilne funkcije na učni in validacijski množici.

vrednosti cenilne funkcije na učni množici in validacijski množici. Spreminjanje teh vrednosti je prikazano na sliki 5.11.



Slika 5.11: Prikaz padanja napake modela pri učenju na večji množici. Za vsak prehod preko 50.000 slik je prikazana vrednost cenilne funkcije na učni množici in validacijski množici.

5.7 Pomen nivojev mreže

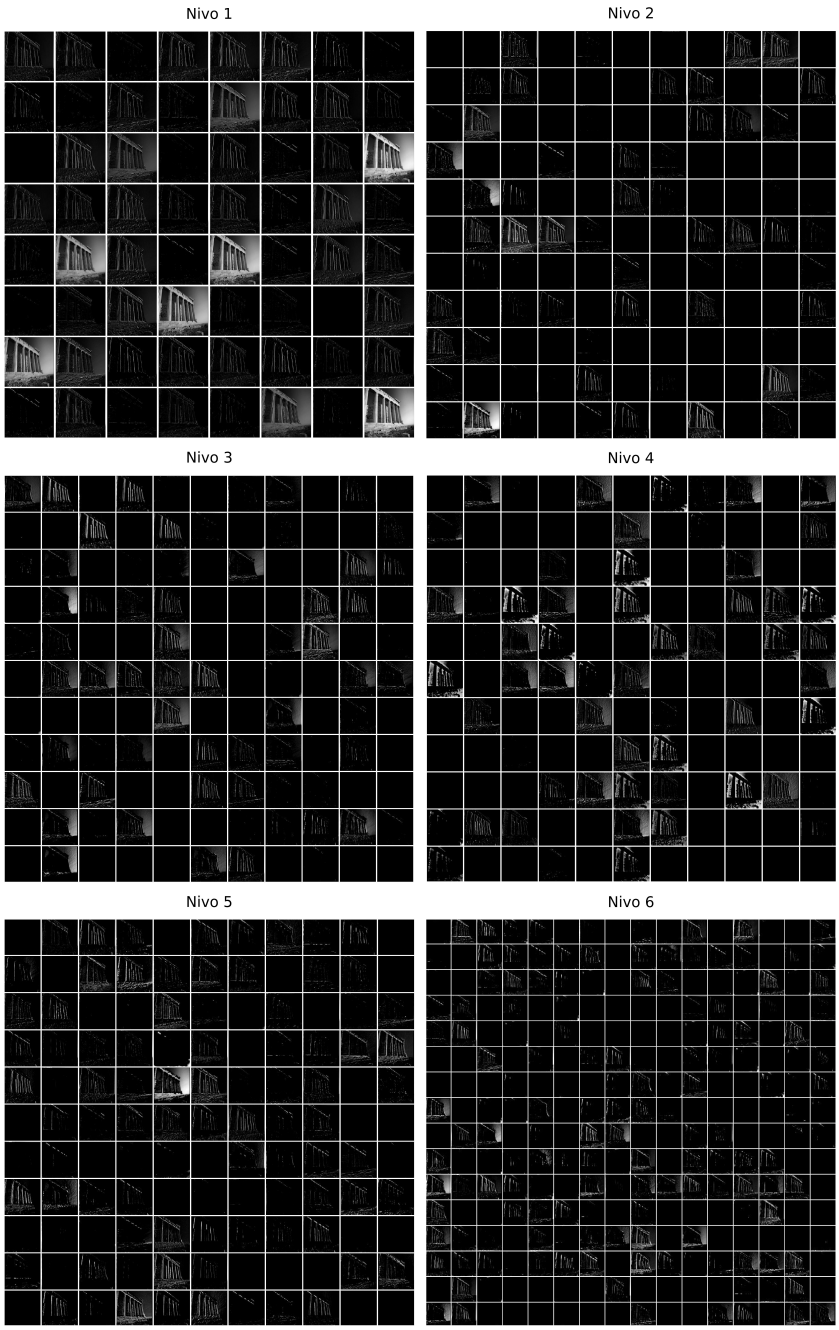
Konvolucijsko nevronsko mrežo si lahko predstavljamo kot nivoje, ki poskrbijo za zajem značilk iz slike. V preteklosti so to počeli z različnimi pristopi, kot so SIFT [39], HOG [39], SURF [40] in ostalimi. Nevronska mreža za to poskrbi sama in izlušči tiste značilke, ki so za določeno nalogo relevantne. Uporabljene značilke lahko v grobem vidimo z vizualizacijo izhodov

konvolucijskih nivojev.

V tem poglavju bomo pogledali, katere značilke zaznajo nivoji. V ta namen smo izrisali izhode prvih šestih konvolucijskih nivojev nevronske mreže, ki so prikazani na sliki 5.12. Vsak nivo predstavlja več slik, ki so izhodi posameznih filtrov, število slik pa je odvisno od števila filtrov. Čeprav ima mreža več nivojev, smo se odločili, da prikažemo le prvih šest, saj so ti najbolj razumljivi. Za vizualizacijo smo si izbrali globalni regresijski pristop z globalno mrežo, saj je ta najbolj zgovorna za vizualizacijo.

Slike prikažejo na katere elemente se filtri v posameznem nivoju odzivajo, oziroma kaj zaznavajo. V prvem nivoju lahko opazimo, da se v večji meri osredotočajo na robove v sliki. Opazimo lahko, da se nekateri odzovejo tudi na večji del slike, kar vidimo kot nekoliko okrnjeno vhodno sliko. V drugem nivoju opazimo že bolj osredotočene odzive. Nekateri še vedno zaznajo robove, drugi pa se že osredotočijo samo na določene dele slike, na primer vodoravne ali navpične robove. Nekateri zaznajo tudi že dele, ki se enako obarvajo, na primer nebo v ozadju.

V tretjem in četrtem nivoju filtri še v večji meri zaznavajo določene dele ali motive. Nekatere značilke je že težje razumeti in opisati. Še vedno je veliko filtrov, ki zaznavajo robove in površine, ki bodo kasneje enako obarvani. To se še vedno nadaljuje tudi v petem in šestem nivoju, kjer je še več značilk, ki imajo pomen za nevronsko mrežo, težje pa so razložljive nam ljudem. Opazimo lahko tudi, da se v kasnejših nivojih pojavlja tudi več izhodov z zelo malo ali praktično nič aktivacijami. Za te predvidevamo, da služijo drugačnim značilkam, ki jih v dani sliki ni bilo mogoče zaznati.



Slika 5.12: Vizualizacija izhodov prvih 6 konvolucijskih nivojev nevronske mreže na primeru slike Atenske akropole. Slika prikazuje, kaj v sliki zaznajo posamezni nivoji in posamezni filtri.

Poglavje 6

Zaključek

V okviru magistrskega dela smo podrobno raziskali področje barvanja črnobelih slik in videov. Ugotovili smo, da so regresijski pristopi bolj natančni, vendar barve niso tako nasičene in so dostikrat bolj sprane. Pristopi s klasifikacijo v osnovi barvajo z močnejšimi in bolj nasičenimi barvami, vendar je barvanje večinoma manj natančno. V primerjavi z obstoječimi pristopi, se je izkazalo, da je naš globalni pristop z regresijo skoraj tako natančen kot pristop Iizuka in sod., ki se je sicer izkazal za najboljšega. Implementacija naših pristopov se nahaja v GitHubu PrimozGodec/ImageColorization[1].

Pri kakovosti barvanja z različnimi pristopi je prisotna velika korelacija, kar pomeni, da je kakovost barvanja zelo odvisna od motiva na sliki. Za primer lahko vzamemo slike z motivom narave, ki se barvajo mnogo bolje kot tiste z manj običajnimi motivi. Slike, ki od te korelacije najbolj odstopajo, so slike, kjer je težje prepoznati teksturo, vendar jo nekateri pristopi še vedno dobro prepoznavajo. Primer take teksture je nebo brez oblakov. Pri testiranju na večji učni množici smo ugotovili, da ta ne spremeni dosti razmerja v kakovosti barvanja med pristopi, je pa pri vseh pristopih opazno izboljšanje kvalitete barvanja tako pri opazovanju napake kot tudi na slikah.

V okviru magistrske naloge smo implementirali več lokalnih pristopov, ki imajo drugačno strategijo barvanja od že obstoječih. Ti pristopi barvajo

[1] `https://github.com/PrimozGodec/ImageColorization`

slike po delih. Kljub temu, da je natančnost teh pristopov v primerjavi s primerljivimi pristopi iz sorodnih del nekoliko slabša, ta pristop omogoča boljše barvanje slik velikosti, ki so različne od tistih, na katerih so bile naučene. Lokalni pristopi so zaradi manjše zahtevnosti povprečno tudi trikrat hitreje naučeni, saj uporabljamo manjše tenzorje, deli slik pa prispevajo dovolj informacij.

V prihodnosti bi se radi posvetili izboljšanju pristopov barvanja videa. Barvanja videa so se lotili že avtorji pristopov za barvanje slik[2], vendar se pri teh opazijo hitri preskoki med različnimi odtenki, ki so posledica barvanja posamezne slike v videu. Problem bi poskusili rešiti z upoštevanjem sosednjih slik. Izdelali bi tudi spletno aplikacijo, ki omogoča barvanje poljubnih slik uporabnikov.

[2]Primeri barvanja videov so dostopni na `http://hi.cs.waseda.ac.jp/~iizuka/projects/colorization/en/` in `http://richzhang.github.io/colorization/`.

Literatura

[1] A. Levin, D. Lischinski, in Y. Weiss, "Colorization using optimization," v *ACM Transactions on Graphics (TOG)*, vol. 23, pp. 689–694, ACM, 2004.

[2] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, in J.-L. Wu, "An adaptive edge detection based colorization algorithm and its applications," v *Proceedings of the 13th annual ACM International Conference on Multimedia*, pp. 351–354, ACM, 2005.

[3] M. Koleini, S. A. Mobadjemi, in P. Moallem, "Automatic Black and White Film Colorization Using Texture Features and Artificial Neural Networks," *Journal of the Chinese Institute of Engineers*, vol. 33, no. 7, pp. 1049–1057, 2010.

[4] E. Reinhard, M. Adhikhmin, B. Gooch, in P. Shirley, "Color transfer between images," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, 2001.

[5] Y.-W. Tai, J. Jia, in C.-K. Tang, "Local color transfer via probabilistic segmentation by expectation-maximization," v *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 747–754, IEEE, 2005.

[6] Z. Cheng, Q. Yang, in B. Sheng, "Deep Colorization," v *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 415–423, 2015.

[7] A. Deshpande, J. Rock, in D. Forsyth, "Learning large-scale automatic image colorization," v *Proceedings of the IEEE International Conference on Computer Vision*, vol. 11-18-Dece, pp. 567–575, 2016.

[8] R. Zhang, P. Isola, in A. A. Efros, "Colorful Image Colorization," v *Computer Vision – ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, pp. 649–666, Cham: Springer International Publishing, 2016.

[9] G. Larsson, M. Maire, in G. Shakhnarovich, "Learning Representations for Automatic Colorization," *arXiv preprint arXiv:1603.06668*, 2016.

[10] S. Iizuka, Edgar Simo-Serra, in H. Ishikawa, "Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification," v *SIGGRAPH '16*, vol. 35, p. 110, ACM, 2016.

[11] P. Nishad in R. Manicka Chezian, "Various Colour Spaces and Colour Space Conversion," *Journal of Global Research in Computer Science*, vol. 4, no. 1, pp. 44–48, 2013.

[12] L. Prangnell, "Visible Light-Based Human Visual System Conceptual Model," *arXiv preprint arXiv:1609.04830*, vol. abs/1609.0, 2016.

[13] S. Bansal in D. Aggarwal, "Color Image Segmentation Using CIELab Color Space Using Ant Colony Optimization," *International Journal of Computer Applications*, vol. 29, no. 9, pp. 28–34, 2011.

[14] K. Jack, "Color spaces," v *Video Demystified - A Handbook for the Digital Engineer*, pp. 15–34, Elsevier, 1989.

[15] I. L. Weatherall in B. D. Coombs, "Skin Color Measurements in Terms of CIELAB Color Space Values," *Journal of Investigative Dermatology*, vol. 99, pp. 468–473, oct 1992.

[16] C. Connolly in T. Fleiss, "A study of efficiency and accuracy in the transformation from RGB to CIELAB color space," *IEEE Transactions on Image Processing*, vol. 6, pp. 1046–1048, jul 1997.

[17] N. Ohta in A. R. Robertson, "CIE Standard Colometric System," v *Colorimetry: Fundamentals and Applications*, pp. 63–114, 2006.

[18] R. Collobert in J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," v *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[19] Y. LeCun, Y. Bengio, in Others, "Convolutional networks for images, speech, and time series," *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10, p. 1995, 1995.

[20] A. Krizhevsky, I. Sutskever, in G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," v *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, in K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[21] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, in L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[22] M. Abadi, A. Agarwal, *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[23] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[24] F. Seide in A. Agarwal, "Cntk: Microsoft's open-source deep-learning toolkit," v *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), pp. 2135–2135, ACM, 2016.

[25] D. P. Kingma in J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] J. M. Joyce, "Kullback-leibler divergence," v *International Encyclopedia of Statistical Science*, pp. 720–722, Springer, 2011.

[27] S. Mannor, D. Peleg, in R. Rubinstein, "The cross entropy method for classification," v *Proceedings of the 22nd international conference on Machine learning - ICML '05*, (New York, New York, USA), pp. 561–568, ACM Press, 2005.

[28] K. Simonyan in A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, sep 2014.

[29] V. Dumoulin in F. Visin, "A guide to convolution arithmetic for deep learning," *Xiv preprint arXiv:1603.07285*, mar 2016.

[30] S. Wu, S. Zhong, in Y. Liu, "Deep residual learning for image steganalysis," *Multimedia Tools and Applications*, pp. 1–17, 2017.

[31] D. Saupe, R. Hamzaoui, in H. Hartenstein, "Fractal Image Compression - An Introductory Overview," Teh. Poročilo, Albert-Ludwigs University at Freiburg, 1997.

[32] S. T. Welstead, "Comparison of Fractal and Wavelet Image Compression," v *Fractal and wavelet image compression techniques*, vol. 40, pp. 155–156, Spie Press, 1999.

[33] C. Wu in J. Chen, "Sampling and Experimental Design," Teh. Poročilo, Department of Statistics and Actuarial Science University of Waterloo, Belgium, 2006.

[34] J. Hauke in T. Kossowski, "Comparison of values of Pearson's and Spearman's correlation coefficients on the same sets of data," *Quaestiones geographicae*, vol. 30, no. 2, p. 87, 2011.

[35] F. Wickelmaier, "An introduction to MDS," *Sound Quality Research Unit, Aalborg University, Denmark*, 2003.

[36] J. Demšar, T. Curk, A. Erjavec, Č. Gorup, T. Hočevar, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, in Others, "Orange: data mining toolbox in Python," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2349–2353, 2013.

[37] S. Ramaswamy, R. Rastogi, in K. Shim, "Efficient algorithms for mining outliers from large data sets," v *Proceedings of the 2000 ACM SIGMOD international conference on Management of data - SIGMOD '00*, pp. 427–438, 2000.

[38] D. Hunter, "Algorithms for Generalised Bradley-Terry Models," *Annals of Statistics*, vol. 32, no. 1, pp. 384–406, 2004.

[39] Y. Ke in R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," v *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, pp. II–506–II–513, IEEE, 2004.

[40] H. Bay, T. Tuytelaars, in L. Van Gool, "SURF: Speeded up robust features," *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I*, pp. 404–417, 2006.

Dodatek A

Spearmanova korelacija rangov med pristopi

Tabela A.1: Tabela prikazuje vrednosti Spearmanove korelacije rangov med pristopi opisanimi v tem delu. Podrobnosti so opisane v poglavju 5.1.

	Zhang in sod.	Iizuka in sod.	Dahl	Reg. lokalna	Reg. lokalna - brez softmax	Reg. lokalna - brez globalne	Reg. globalna	Reg. globalna - brez globalne	Reg. globalna VGG	Klas. brez uteži - S+G	Klas. brez uteži - D+G	Klas. z utežmi D+G	Klas. z utežmi - S+G
Zhang	1,00	0,86	0,86	0,86	0,84	0,88	0,87	0,88	0,89	0,90	0,94	0,92	0,90
Iizuka	0,86	1,00	0,89	0,94	0,94	0,90	0,94	0,92	0,93	0,90	0,85	0,85	0,88
Dahl	0,86	0,89	1,00	0,90	0,90	0,98	0,88	0,95	0,91	0,86	0,88	0,87	0,84
R. lok.	0,86	0,94	0,90	1,00	0,94	0,90	0,93	0,91	0,93	0,90	0,86	0,85	0,88
R. lok. brez sm.	0,84	0,94	0,90	0,94	1,00	0,91	0,93	0,91	0,92	0,87	0,84	0,83	0,86
R. lok. brez gl.	0,88	0,90	0,98	0,90	0,91	1,00	0,90	0,96	0,92	0,87	0,89	0,88	0,85
R. glob.	0,87	0,94	0,88	0,93	0,93	0,90	1,00	0,91	0,93	0,90	0,86	0,86	0,89
R. glob. brez gl.	0,88	0,92	0,95	0,91	0,91	0,96	0,91	1,00	0,93	0,87	0,86	0,85	0,85
R. glob. VGG	0,89	0,93	0,91	0,93	0,92	0,92	0,93	0,93	1,00	0,90	0,88	0,88	0,89
K. brez ut. S+D	0,90	0,90	0,86	0,90	0,87	0,87	0,90	0,87	0,90	1,00	0,92	0,92	0,94
K. brez ut. G+D	0,94	0,85	0,88	0,86	0,84	0,89	0,86	0,86	0,88	0,92	1,00	0,98	0,91
K. ut. G+D	0,92	0,85	0,87	0,85	0,83	0,88	0,86	0,85	0,88	0,92	0,98	1,00	0,92
K. ut. S+D	0,90	0,88	0,84	0,88	0,86	0,85	0,89	0,85	0,89	0,94	0,91	0,92	1,00

Dodatek B

Implementacije globokih mrež

B.1 Arhitektura S+G

Arhitektura S+G predstavljena z implementacijo v vmesniku Keras:

```
""" Glavna mreža """
num_classes = 400
main_input = Input(shape=input_shape,name='image_part_input')

x = Conv2D(64, (3, 3), strides=(2, 2), padding="same",
           activation="relu")(main_input)
x = Conv2D(128, (3, 3), padding="same", activation="relu")(x)

x = Conv2D(128, (3, 3), strides=(2, 2), padding="same",
           activation="relu")(x)
x = Conv2D(256, (3, 3), padding="same", activation="relu")(x)

x = Conv2D(256, (3, 3), strides=(2, 2), padding="same",
           activation="relu")(x)
x = Conv2D(512, (3, 3), padding="same", activation="relu")(x)

x = Conv2D(512, (3, 3), padding="same", activation="relu")(x)
main_output = Conv2D(256, (3, 3), padding="same", activation="relu")(x)

""" Dodatni nivo k globalni mreži """
```

```
vgg16 = VGG16(weights="imagenet", include_top=True)
vgg_output = Dense(256, activation='softmax',
    name='predictions')(vgg16.layers[-2].output)

""" Združevanje glavne in globalne mreže """
def repeat_output(input):
    shape = K.shape(x)
    return K.reshape(K.repeat(input, 4 * 4), (shape[0], 4, 4, 256))

vgg_output = Lambda(repeat_output)(vgg_output)

# zamrzovanje nivojev mreže VGG16
for layer in vgg16.layers:
    layer.trainable = False

""" Združena mreža """
merged = concatenate([vgg_output, main_output], axis=3)

last = Conv2D(256, (3, 3), padding="same")(merged)

last = UpSampling2D(size=(2, 2))(last)
last = Conv2D(256, (3, 3), padding="same", activation="relu")(last)
last = Conv2D(256, (3, 3), padding="same", activation="relu")(last)

last = UpSampling2D(size=(2, 2))(last)
last = Conv2D(256, (3, 3), padding="same", activation="relu")(last)
last = Conv2D(400, (3, 3), padding="same", activation="relu")(last)

def resize_image(x):
    return K.resize_images(x, 2, 2, "channels_last")

# večdimenzionalni softmax
def custom_softmax(x):
    sh = K.shape(x)
    x = K.reshape(x, (sh[0] * sh[1] * sh[2], num_classes))
    x = K.softmax(x)
    x = K.reshape(x, (sh[0], sh[1], sh[2], num_classes))
    return x
```

```
last = Activation(custom_softmax)(last)
last = Lambda(resize_image)(last)
```

B.2 Arhitektura D+G

Arhitektura D+G predstavljena z implementacijo v vmesniku Keras:

```
""" Glavna mreža """
main_input = Input(shape=input_shape, name='image_part_input')

x = Conv2D(64, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01),
           name="conv1")(main_input)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
x1 = Conv2D(128, (3, 3), padding="same", activation="relu",
            kernel_regularizer=regularizers.l2(0.01), name="conv2")(x)
x = Conv2D(128, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01), name="conv3")(x1)
x = Conv2D(128, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01), name="conv4")(x)
x = add([x, x1])

x = Conv2D(128, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01), name="conv5")(x)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
x1 = Conv2D(256, (3, 3), padding="same", activation="relu",
            kernel_regularizer=regularizers.l2(0.01), name="conv6")(x)
x = Conv2D(256, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01), name="conv7")(x1)
x = Conv2D(256, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01), name="conv8")(x)
x = add([x, x1])

x = Conv2D(256, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01))(x)
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
```

```
x1 = Conv2D(512, (3, 3), padding="same", activation="relu",
            kernel_regularizer=regularizers.l2(0.01))(x)
x = Conv2D(512, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01))(x1)
x = Conv2D(512, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01))(x)
x = add([x, x1])

x = Conv2D(512, (3, 3), padding="same", activation="relu",
           kernel_regularizer=regularizers.l2(0.01))(x)
main_output = Conv2D(256, (3, 3), padding="same", activation="relu",
    kernel_regularizer=regularizers.l2(0.01))(x)

""" Dodatni nivo k globalni mreži """
vgg16 = VGG16(weights="imagenet", include_top=True)
vgg_output = Dense(256, activation='relu',name='predictions')(
    vgg16.layers[-2].output)

""" Združevanje glavne in globalne mreže """
def repeat_output(input):
    shape = K.shape(x)
    return K.reshape(K.repeat(input, 28 * 28), (shape[0], 28, 28, 256))

vgg_output = Lambda(repeat_output)(vgg_output)

# zamrzovanje nivojev mreže VGG16
for layer in vgg16.layers:
    layer.trainable = False

merged = concatenate([vgg_output, main_output], axis=3)

""" Združena mreža """
last = Conv2D(128, (3, 3), padding="same")(merged)

last = Conv2DTranspose(
    64, (3, 3), strides=(2, 2), padding="same", activation="relu",
    kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(64, (3, 3), padding="same", activation="relu",
```

```
                kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(64, (3, 3), padding="same", activation="relu",
                kernel_regularizer=regularizers.l2(0.01))(last)

last = Conv2DTranspose(
    64, (3, 3), strides=(2, 2), padding="same", activation="relu",
    kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(32, (3, 3), padding="same",activation="relu",
                kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(2, (3, 3), padding="same",activation="relu",
                kernel_regularizer=regularizers.l2(0.01))(last)


def resize_image(x):
    return K.resize_images(x, 2, 2, "channels_last")

last = Lambda(resize_image)(last)
```

B.3 Arhitektura X-VGG

Arhitektura X-VGG predstavljena z implementacijo v vmesniku Keras:

```
""" Mreža VGG16"""
vgg16 = VGG16(weights="imagenet", include_top=False,
                input_shape=input_shape)

# zamrzovanje nivojev mreže VGG16
for layer in vgg16.layers:
    layer.trainable = False

""" Dodani nivoji mreže """
last = UpSampling2D(size=(2, 2))(vgg16.output)
last = Conv2D(256, (3, 3), padding="same", activation="relu",
                kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(256, (3, 3), padding="same", activation="relu",
                kernel_regularizer=regularizers.l2(0.01))(last)
```

```
last = UpSampling2D(size=(2, 2))(last)
last = Conv2D(128, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(128, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)

last = UpSampling2D(size=(2, 2))(last)
last = Conv2D(64, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(64, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)

last = UpSampling2D(size=(2, 2))(last)
last = Conv2D(32, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)
last = Conv2D(2, (3, 3), padding="same", activation="relu",
              kernel_regularizer=regularizers.l2(0.01))(last)


def resize_image(x):
    return K.resize_images(x, 2, 2, "channels_last")

last = Lambda(resize_image)(last)
```

Dodatek C

Primerjava pristopov s spletno anketo

Tabela C.1: Razporeditev rezultatov evalvacije s spletno anketo. Števila prikazujejo kolikokrat so anketiranci izbrali pristop v vrstici za boljšega od tistega v stolpcu.

	Iizuka in sod.	Dahl	Reg. lokalna	Reg. globalna	Reg. globalna VGG	Klas. brez uteži - S+G	Klas. z utežmi - S+G
Iizuka		700	765	642	716	743	796
Dahl	339		551	395	488	601	647
Regresija lokalna	271	487		346	501	518	622
Regresija globalna	387	635	688		635	656	719
Regresija globalna VGG	314	543	543	394		599	687
Klasifikacija brez uteži S+G	298	431	518	370	428		561
Klasifikacija z utežmi S+G	241	385	417	312	343	476	